

AD-755 393

SLAMCODE: FINITE ELEMENT STRESS WAVE
ANALYSIS

D. W. McCowan

Teledyne Geotech

Prepared for:

Air Force Office of Scientific Research

14 September 1972

DISTRIBUTED BY:

NTIS

National Technical Information Service
U. S. DEPARTMENT OF COMMERCE
5285 Port Royal Road, Springfield Va. 22151

**BEST
AVAILABLE COPY**

AL-72-1



.....contributing to man's
understanding of the environment world

31 OCT 1972

SLAMCODE: FINITE ELEMENT STRESS WAVE ANALYSIS

BY

D.W. McCOWAN

SPONSORED BY

ADVANCED RESEARCH PROJECTS AGENCY

MONITORED BY

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH

RPA ORDER NO. 1357

14 SEPTEMBER 1972



 **TELEDYNE GEOTECH**
ALEXANDRIA LABORATORIES

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
U S Department of Commerce
Springfield VA 22151

Approved for public release;
distribution unlimited.

60
R

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)

Teledyne Geotech
Alexandria, Virginia

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2c. GROUP

3. REPORT TITLE

SLAMCODE: FINITE ELEMENT STRESS WAVE ANALYSIS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

ScientificInterim

5. AUTHOR(S) (Last name, first name, initial)

McCowan, D.W.

6. REPORT DATE

14 September 1972

7a. TOTAL NO. OF PAGES

60

7b. NO. OF REFS

3

8a. CONTRACT OR GRANT NO.

F44620-69-C-0082

b. PROJECT NO.

AO 1357

c. 62701D

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

AL-72-1

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

AFOSR - TR - 73 - 0120

10. AVAILABILITY/LIMITATION NOTICES

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES

TECH, OTHER

12. SPONSORING MILITARY ACTIVITY

Air Force Office of Scientific Res.
1400 Wilson Boulevard NPG
Arlington, Virginia

13. ABSTRACT

The program SLAMCODE developed in this study and described in this Volume is a large Finite Element Method (FEM) code capable of dynamic analysis of transient problems. It was adapted from the original SLAMCODE written by C. J. Costantino (Costantino, 1968) by incorporating the implicit Newmark B-integration (Newmark, 1959) and by constraining the free surface to be stress free. The present code also differs from the original by not permitting plastic elements. It calculates solutions for axisymmetric, plane stress or plane strain problems, as in the original code. It is designed to run on the IBM 360 family of computers under OS/360. Being written in Fortran, it can be easily adapted to other computer systems. In fact, it was written with this requirement in mind by minimizing the number of system-dependent Fortran peculiarities.

14. KEY WORDS

Finite Element
Stress Waves

1

Unclassified

Security Classification

SLAMCODE: FINITE ELEMENT STRESS WAVE ANALYSIS

ALEXANDRIA LABORATORIES REPORT NO. AL-72-1

Effective Date of Contract:	1 March 1969
Contract Expiration Date:	31 March 1972
Amount of Contract Dollars:	\$ 590,572
Program Code:	1F10
Contract Number:	F-44620-69-C-0082
ARPA Order No.:	1357
Principal Investigator:	Carl A. Newton
Period Covered:	1 March 1969 through 31 March 1972



TABLE OF CONTENTS

	Page No.
SLAMCODE Computer Program	1
REFERENCES	5
APPENDICES	
APPENDIX A	
SLAMCODE SOURCE LISTING	
APPENDIX A1	
(MINBND) SOURCE LISTING	
APPENDIX B	
ØS/360 SLAMCODE LINK EDITOR CONTROL CARDS	
APPENDIX C	
ØS/360 FORTRAN H SLAMCODE JOB CONTROL LANGUAGE CARDS	
a. Requires two 9-track tape drives	
b. Requires four 9-track tape drives	

LIST OF TABLES

Table Title	Table No.
Stress-Constrained SLAMCODE Parameter Limits	1
360 Stress-Constrained SLAMCODE Run Chart	2
Card Input Chart for 360 Stress-Constrained SLAMCODE	3
Data Set Chart for 360 Stress-Constrained SLAMCODE	4

FIGURE TITLE

Figure Title

Page No.

SLAMCODE overlay structure. Brackets
indicate subroutines.

2

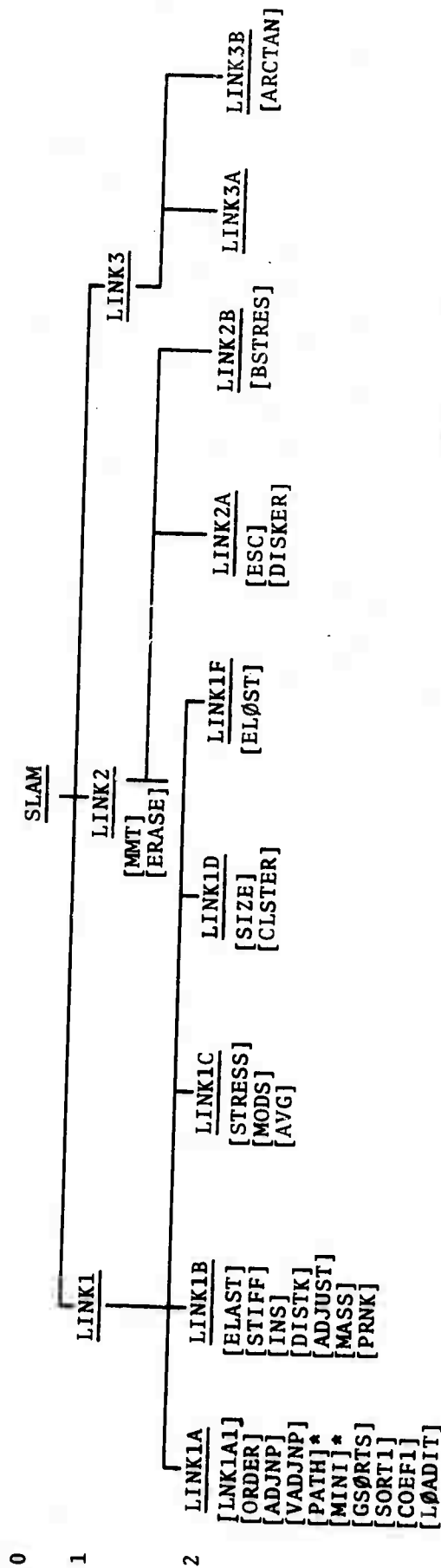
SLAMCODE Computer Program

The program SLAMCODE developed in this study and described in this Volume is a large Finite Element Method (FEM) code capable of dynamic analysis of transient problems. It was adapted from the original SLAMCODE written by C. J. Costantino (Costantino, 1968) by incorporating the implicit Newmark β -integration (Newmark, 1959) and by constraining the free surface to be stress free. The present code also differs from the original by not permitting plastic elements. It calculates solutions for axisymmetric, plane stress or plane strain problems, as in the original code. It is designed to run on the IBM 360 family of computers under OS/360. Being written in Fortran, it can be easily adapted to other computer systems. In fact, it was written with this requirement in mind by minimizing the number of system-dependent Fortran peculiarities.

The Fortran source cards are listed in Appendix A. The Link Editor control cards are listed in Appendix B. These control cards are appended to the object module for execution of the program. Appendix C contains examples of Job Control Language cards for the Fortran H computer using either two or four 9-track tape drives. Tables 1-4 describe the operation of the program and Figure 1 is a block diagram of the overlay structure. Finally, detailed descriptions of the input parameters and operation of the program can be gleaned by reading the comments in the source listing. All input data is described

Level

7



*[PATH] and [MINI] may be replaced by Rosen's matrix bandwidth minimization..

Figure 1. SLAMCODE overlay structure. Brackets indicate subroutines.

where it is read in the program listing.

The bandwidths of the diagonal matrices are controlled by the numbering of the nodal points. The bandwidth minimization algorithms listed in Appendix A are those used by Costantino, (PATH) and (MINI). A third algorithm written by Rosen (1968) and adapted for SLAMCODE is listed in Appendix A1, (MINBND). The listing of the original Rosen algorithm contained some typographical errors which were corrected for (MINBND), but the comment cards which we have not reproduced in Appendix A1 are very useful. (PATH) is much faster than (MINBND) but its effectiveness depends upon the selection of starting node points. (MINI) has not decreased the bandwidth for any of our element arrays.

In general, large problems are solved by several runs of the program, all output being saved on magnetic tape or cataloged disk files. A wide range in input/output (I/O) flexibility is available with OS/360 and this should be exploited to maximize program efficiency. For example, tape units should only be used in preference to disks when the time reduction made possible by saving intermediate results exceeds the time lost by the tape drives.

The program in its present form is I/O bound, i.e., a large amount of the computer time is spent spinning tapes and disks. In a typical 360/67 run there are approximately the same number of I/O seconds as there are CPU seconds. This performance

might be improved on larger machines where I/O buffers and program dimensions could be increased. The 360/67 version runs in a 280K byte region of core memory.

The number of I/O devices required depends on the operating system. The 360/67 system can assign many files to one device. A general rule of thumb for a system with 280K bytes of available core memory is that the program requires at least four tape drives and a hundred cylinders of disk file space.

REFERENCES

- Costantino, C. J., 1968, Stress Waves in Layered Arbitrary Media, Final Report to Space and Missile Systems Organization (SAMSO) Norton Air Force Base, California: SAMSO TR 68-181.
- Newmark, M., 1959, A Method of Computation for Structural Dynamics, J. of Engineering Mechanics Division, ASCE, 85, EM 3, 67-94.
- Rosen, R., 1968, Matrix Bandwidth Minimization, Proceedings of 23d National Conference, ACM, Brandon/Systems Press, Inc., New Jersey, pp 585-595.

Table 1
STRESS-CONSTRAINED SLAMCODE PARAMETER LIMITS

MACHINE: 360/67			
WORD LENGTH:	FLOATING POINT: 8 BYTES = 64 BITS		
	FIXED POINT: 4 BYTES = 32 BITS		
CYCLE TIME:	0.75 MICROSEC.		
MEMORY SIZE:	280K BYTES = 35K FLOATING POINT WORDS		
PARAMETER	DESCRIPTION	MAXIMUM VALUE	COMMENT
NUMNP	NUMBER OF NODE POINTS	1000	
NUMEL	NUMBER OF ELEMENTS	1250	
--	NUMBER OF ADJACENT NODES TO ANY NODE POINT	8	NEVER ACTUALLY SET BUT IMPLIED IN A DIMENSION STATEMENT
LINE	NUMBER OF STRESSED LINES	1	<6 FOR STRESSED NODES
LOADNP	NUMBER OF NODES PER STRESSED LINE	50	
NZONES	NUMBER OF MATERIAL ZONES	20	
NUMST	NUMBER OF STARTING NODES	20	
--	MATRIX BLOCK SIZE=MESH BANDWIDTH+1 NODES	30	LINK1 LIMIT ON BANDWIDTH IS 100; HOWEVER, LINK2 LIMIT IS 29
--	PLASTIC ELEMENT TAPE BLOCKING FACTOR	26	NOT A CONSIDERATION IN THIS VERSION OF SLAM
--	NUMBER OF GRAND PARTITIONS IN PATH PROCEDURE	100	PARAMETER INVOLVED IN BANDWIDTH MINIMIZATION
--	NUMBER OF BLOCKS OF NODES ON STIFF./STRESS TAPE	100	e.g., MINIMUM BANDWIDTH FOR 1000 NODE PROBLEM = 9
NUMOUT	NUMBER OF OUTPUT NODES	100	

Table 2
360 STRESS-CONSTRAINED BLA CODE RUN CHART

KRUN	CARD GROUPS SKIPPED	OUTPUT SKIPPED	TAPE ASSIGNMENTS		LINK 2 RUN TIME	LINK 3 RUN TIME	DESCRIPTION & COMMENTS
0	NONE	NONE	STIFF.&STRESS TAPE OUT ON #10 UTR TAPE OUT ON #20 IF TMAX #0 DTH TAPE OUT ON #8 IF TMAX #0 OTH TAPE OUT ON #3 IF TMAX #0	0 - TMAX	TSTRESS - TMAX	INITIAL RUN THERUN MUST BE 0 MESH TITLE WRITTEN ON STIFF & STRESS,UTR,DTH,& OTH TAPES	
1	NONE	ALL LINK1 STIFF, MASS & STRESS TABLES	STIFF.&STRESS TAPE IN ON #10 UTR TAPE OUT ON #20 DTH TAPE OUT ON #8 OTH TAPE OUT ON #3 IF TMAX #0	0 - TMAX	TSTRESS - TMAX	LINK1 RERUN THERUN MUST BE 0 PROBLEM TITLE WRITTEN ON UTR,DTH,& OTH TAPES	
2	ALL LINK1 INPUT 1.1 - 1.6	ALL LINK1 OUTPUT	STIFF.&STRESS TAPE IN ON #10 UTR TAPE IN ON #20 DTH TAPE IN/OUT ON #8 OTH TAPE OUT ON #3	THERUN-TMAX	TSTRESS-TMAX	LINK 2 RERUN	
3	ALL LINK1 INPUT 1.1 - 1.6	ALL LINK1 OUTPUT LINK3 STRESS OUT- PUT	STIFF.&STRESS TAPE IN ON #10 UTR TAPE IN ON #20 DTH TAPE IN/OUT #8 OTH TAPE OUT ON #3	THERUN - TMAX	TSTRSS - TMAX	LINK2 SUMMARY OUTPUT SUMMARY=NO LINK3 STRESS OUTPUT	
4	ALL LINK1 INPUT ALL LINK2 INPUT 1.1-2.2	ALL LINK1 OUTPUT ALL LINK2 OUT- PUT LINK3 STRESS OUT- PUT	STIFF.&STRESS TAPE IN ON #10 DTH TAPE IN ON #8 OTH TAPE OUT ON #3	NOT RUN	TSTRSS - TMAX	LINK3 SUMMARY OUTPUT	
5	ALL LINK1 INPUT ALL LINK2 INPUT 1.1-2.2	ALL LINK1 OUTPUT ALL LINK2 OUT- PUT	STIFF.&STRESS TAPE IN ON #10 DTH TAPE IN ON #8 OTH TAPE OUT ON #3	NOT RUN	TSTRSS - TMAX	LINK3 RERUN	

Table 3

CARD INPUT CHART FOR 360 STRESS-CONSTRAINED SLAMCODE

CARD GROUP	READING ROUTINE	COND. OF READ	READ LIST	FORMAT	PARAMETERS
0.1	SLAM		KRUN, ANAME	(15, 9A8)	- RUN TYPE SWITCH - 72 CHAR. RUN TYPE IDENTIFIER
1.1	LNK1A1		TITLE NUMNP, NUMEL, ISTRES, IPRINT	(10A8) (415)	- MESH TITLE/PROBLEM TITLE - # NODES - # ELEMENTS - GEOMETRY SWITCH - PRINT SWITCH
1.2	LNK1A1	NPN < NUMNP	ANAME NPN, R(NPN), Z(NPN), ITYPE(NPN), THETA(NPN)	(10A8) (15, 2E10.4, I10, E10.4)	- CARD GROUP IDENTIFIER (NOT PRINTED) - NODE # - RADIUS - DEPTH - NODE TYPE - ROLLER ANGLE
1.3	LNK1A1	I < LINES	ANAME LINES, (LØADNP(I), I=1, LINES) (NPLØAD(I, J), SNØRM(I, J), J=1, LØADNP(I))	(10A8) (1415) (15, E10.2)	- ADAPING TYPE IDENTIFIER - STRESSED LINES - STRESSED LINES ON LINE I - NODE# of Jth NODE ON STRESSED LINE I - SNØRM(I, J) - SURFACE NORMAL ANGLE
1.4	LNK1A1	IZ < LINES	ANAME NZONES IZ, ANAMF IELAST, IPLAST, WGT, E1, E2, E3, E4, E5	(10A8) (15) (15, 9A8) (215, 6E10.4)	- CARD GROUP IDENTIFIER (NOT PRINTED) - MATERIAL ZONES - ZONE # - ZONE MATERIAL IDENTIFIER - ELASTICITY SWITCH - PLASTICITY SWITCH - DENSITY - ELASTIC MODULUS - POISSON'S RATIO - ANISOTROPIC PARAMETERS
1.5	LNK1A1	NUME < NUMEL	ANAME NUME, IZONE, NPI, NPJ, NPK, NPL	(10A8) (615)	- CARD GROUP IDENTIFIER (NOT PRINTED) - ELEMENT # - ZONE # - NODE#s OF VERTICES

Table 3 (Cont'd.)

1.6	LNK1A1	ANAME NUMST (NSTART(1), I=1, NUMST)	(10A8) (15) (1415)	- CARD GROUP IDENTIFIER (NOT PRINTED) - #STARTING NODES - STARTING NODE #'S - MAX. RUN TIME FOR PRESENT RUN - RESTART TIME - TIME INCREMENT - DIVISOR FOR CHOOSING DT - INTEGRATION SCHEME PARAMETER - CRITICAL DAMPING FOR ARTIFICIAL VISCOSITY
2.1	L. NK2	TMAX, TRERUN, ET, KDT, KINT, BETA, PDAMP	(3E10.0, 215, 2E10.0)	ANAME NUMST NSTART TMAX TRERUN ET KDT KINT BETA PDAMP
2.2	BSTRES	IPULSE, IDIREC, N1, N2, STAMP	(415, E10.2)	IPULSE IDIREC N1 N2 STAMP
3.1	LINK3A	TSTRSS, TMAX, NUMOUT, IØUT, JØUT (NPØUT(1), I=1, NUMØUT)	(2E10.0, 315) (1415)	TSTRSS TMAX NUMØUT IØUT JØUT NPØUT - STRESS PULSE TYPE SWITCH - STRESS DIRECTION SWITCH - FIRST STRESSED NODE - LAST STRESSED NODE - STRESS AMPLITUDE - OUTPUT START TIME - OUTPUT STOP TIME - # OUTPUT NODES - # INTEGRATION POINTS/PRINTED OUTPUT - # INTEGRATION POINTS/TAPE OUTPUT - OUTPUT NODES #'S

Table 4
DATA SET CHART FOR 360 STRESS-CONSTRAINED SLAMCODE

PROGRAM	DSRN	I/O	DEVICE	SAVED	COMMENT
LINK1A	1 4 8 14	0 0 0 0	SORTED ELEMENT DATA SORTED NODE DATA ADJACENCY TABLES PLASTIC ELEMENT DATA	DISK DISK DISK DUMMY	NEVER NEVER NEVER NEVER
LINK1B	1 3 4 8 12	I 0 I I 0	AS ABOVE UNBLOCKED STIFFNESS TABLES AS ABOVE AS ABOVE LINK1C ELEMENT DATA	DISK DISK DISK DISK	NEVER NEVER NEVER NEVER
LINK1C	3 4 8 12	0 I I I	UNBLOCKED STIFFNESS AND STRESS TABLES AS ABOVE AS ABOVE AS ABOVE	DISK DISK DISK DISK	NEVER NEVER NEVER NEVER
LINK1D	3 10	I I0	AS ABOVE BLOCKED STIFFNESS AND STRESS TAPE	TAPE	AFTER KRUN=0 RUN
LINK1F	12	0	PLASTIC SAVE TAPE	TAPE	
LINK2	8	I0	DISPLACEMENT TIME HISTORY TAPE	TAPE	AFTER KRUN=0(TMAX#0)
LINK2A	10 20 22	I 0 I0	AS ABOVE UPPER TRIANGULARIZATION TAPE NON-SEQUENTIAL SCRATCH	TAPE DISK	AFTER KRUN=0(TMAX#0) AND KRUN=1 RUNS NEVER
LINK2B	8	I0	AS ABOVE	TAPE	AFTER KRUN=0(TMAX#0) 1(TMAX#0), 2, 3 RUNS
LINK3	8	I	AS ABOVE		
LINK3A	10	I	AS ABOVE		
LINK3B	3 8	0 I	OUTPUT TIME HISTORY TAPE AS ABOVE	TAPE	AFTER ALL KRUN

LINK1 SCRATCH
LINK1 SCRATCH
LINK1 SCRATCH
NEVER USED IN THIS VERSION

LINK1 SCRATCH

LINK1 SCRATCH

CONTINUATION OF DATA SET ABOVE

OUTPUT FOR KRUN=0, INPUT FOR KRUN=1

THIS VERSION OF SLAMCODE IS INCAPABLE
OF PLASTIC ANALYSIS; HENCE, LINK1F SHOULD
NOT BE EXECUTED.

WRITES TITLE AND COMMON BLOCK RECORDS
FOR KRUN=0, 1
READS TITLE AND COMMON BLOCK RECORDS
FOR KRUN=2, 3

INPUT FOR KRUN=0, 1
OUTPUT FOR KRUN=0, 1

LINK2A SCRATCH

WRITES PROBLEM INTEGRATION OUTPUT FOR
KRUN=0, 1, 2, 3
CONTINUATION OF DATA SET ABOVE
INPUT FOR KRUN=2, 3

READS TITLE AND COMMON BLOCK RECORDS
FOR ALL KRUN

READS BLOCKED STRESS TABLES FOR ALL
KRUN

PROBLEM OUTPUT FOR ALL KRUN
READS INTEGRATION OUTPUT FOR ALL KRUN

```

PROGRAM SLAM
IMPLICIT REAL*(A-M,N-Z)
COMMON MAXNP,MAXJNP,MAXLINE,MAXLOAD,MAXNPB,MAXELB,NUMNP,MINNEL,
ISTRES,ALINES,NUMPEL,MAXNP(1),PERIOD,OT,NPLOAD(1,50),
RAD(1,50),ZAD(1,50),SNORM(1,50),TITLE(10),MAXRD,
MLOCK,NREADS,MAXWRK,TREAL,DAMP,ACTA,KRUN,
MZONE,NZONE5,TPRINT,NPTN,IELAST,WT,ET,E2,E3,E4,E5
DIMENSION NPTN(1000),IELAST(20),WT(20),F2(20),F3(20),
E4(20),E5(20)
DIMENSION ANAME(9)

```

IBM 360/67 VERSION OF VISCO-ELASTIC SLAM CODE
CONVERTED APRIL, 1971 BY D.W. MCCOMAN WITH CONSULTATION FROM
C.J. CONSTANTINO

UPDATED TO DO IMPLICIT TIME INTEGRATION OCTOBER, 1971
 UPDATED TO CONSTRAIN BOUNDARY STRESS MARCH, 1972

SET LIMITS

CALL NOTIFY
MAXNP=1000
MAXADJP=R
MXLINE=1
MYLOAD=50
MXNPP=100
MAXELB=26
MXZONE=20
MAXMBK=30

MATNIP	=	MAXIMUM	NUMBER OF NODE POINTS
MATNJP	=	MAXIMUM	NUMBER OF NODE POINTS ADJACENT TO ANY NODE
MATNLIN	=	MAXIMUM	NUMBER OF LOADED LINES OR SURFACES
MATNLOAC	=	MAXIMUM	NUMBER OF LOADED NODES PER LOADED LINE
MATNPRA	=	MAXIMUM	NUMBER OF NODE POINTS PER BUFFER
MATNELB	=	MAXIMUM	PLASTIC ELEMENTS PER BLOCK
MATNZONE	=	MAXIMUM	NUMBER OF MATERIAL ZONES
MATMAREK	=	MAXIMUM	BANDWIDTH FOR INTEGRATION SCHEME

REAR AND PRINT RUN SWITCH

```

READ(5,1) KRUN, ANAME
1  FORMAT(15,9A)
WRITE(6,4) KRUN, ANAME
4  FORMAT(14I,5MKRUN=,15/1X,9A)

```

```

KRUN      = SWITCH FOR RERUN
KKRUN     = 0 INITIAL RUN
           = 1 LINK1 RERUN
           = 2 LINK2 RERUN
           = 3 LINK2 SUMMARY RERUN
           = 4 LINK3 SUMMARY RERUN
           = 5 LINK3 RERUN
NAME      = DESCRIPTOR ON DATA CARD, FOR RUN TYPE ONLY

```

```
IF(KRUN.GT.1) GO TO 3
CALL LINK1
IF(KRUN.GT.3) GO TO 5
CALL LINK2
CONTINUE
CALL LINK3
CALL EXIT
STOP
END
```

```

SUBROUTINE LINK1
  IMPLICIT REAL*8(A-H,O-Z)
  COMMON MAXP,MAXCJP,MLXLINE,MLX
  ISTRS=1,MLX,NUMBER
  RAD(1,50),ZAD(1,50)
  MLOCK,NREADS,MAXP
  MXZONE,NZONES,IPRT
  DIMENSION NPTN(1000),IELA
  E4(20),E5(20)

```

```

LINK1 PREPARES THE STIFFNESS, MASS,
FOR THE RUN
CALL LINK1A
(TFTRUN.EQ.1) GO TO 10
CALL LINK1R
CALL LINKIC
CALL LINKID
CALL LINKIN
(TFTRUN.EQ.1, NR, NUNDEL, FO, C) RETURN
CALL LINKIF
RETURN
END

```

```

SUBROUTINE LINK1A
  IMPLICIT REAL*8(A-H,O-Z)
  COMMON MAXNP, MXADJP, MXELINE, MXLOAD, MNPB, MAXELB, NUMNP, NUMEL,
  ISTRS, LINES, NUMPEL, LOADNP(1), PERIOD, QT, NPLOAD(1,50),
  RAD(1,50), ZAD(1,50), SNORW(1,50), TITLE(10), KARRD,
  MELTCK, NPCAOS, MAXMBK, TREAL, POAMP, BETA, KRUN,
  MXZLINE, NTONES, IPRINT, NPTN, TELA ST, WGT, E1, E2, E3, E4, E5
  DIMENSION NPADJ(1000,9), NAQJNP(1000), NADJEL(1000),
  NPJNSTRT(20), NSTART(20), TGP(100), TELAST(20), IPLAST(20),
  EGT(20), E1(20), E2(20), E3(20), E4(20), E5(20), R(1000), Z(1000),
  ITYPE(1000), THETA(1000), TMP(5000), NTMP(8,1250)
  DIMENSION NPTN(1000)
  EQUIVALENCE (TMP(1),NTMP(1,1))

```

THIS LINK READS THE MESH DATA CARDS, RENUMBERS THE MESH,
SORTS THE MESH DATA, AND PREPARES TABLES FOR THE STIFFNESS,
STRESS, AND PLASTIC MATRICES CALCULATIONS

SET MAXIMUM NUMBER OF STARTING NODES

MMYSTRT=20

```

C      READ AND STORE CARO INPUT
C
C      CALL LNKAL(MAXNP,MXLIN,MXLOAD,MXZONE,MUNNP,MUNEL,(PRINT,
11STRES,MUNEL,NZONE,S,R,2,ITYPE,THETA,LINFS,LOADNP,NPLOAO,
2RAD,ZAO,SNORM,IFLAST,IFLAST,MGT,E1,E2,E3,E4,E5,MUMST,NSTART,
3MXSTR,NTP,TITLE)
C
C      ZERO OUT EVERYTHING
C
C      DO 5 I=1,MUNNP
C        NAOJNP(I)=0
C        NAOJEL(I)=0
C      DO 5 J=1,MXADJP
C        NPADJ(I,J)=0
C      5 NPADJ(I,J)=0
C
C      CALCULATE NPAOJ, NAOJNP, NAOJEL
C
C      NPAOJ = ADJACENT NODE POINT NUMBERS ARRAY
C      NAOJNP = NUMBER OF ADJACENT NODE POINTS ARRAY
C      NAOJEL = NUMBER OF ADJACENT ELEMENTS ARRAY
C
C      DO 7 M=1,MUNEL
C      7 CALL ADJNP(MXADJP,MAXNP,MUNNP,NPAOJ,NAOJEL,NTMP(2,M),NTMP(5,M),
11NTMP(6,M),NTMP(7,M),NTMP(8,M))
C      CALL VADJNP(MXADJP,NAOJNP,MAXNP,MUNNP,NPAOJ)
C      PRINT
C
C      IF((PRINT.NE.1).AND.((PRINT.NE.8))) GO TO 4
C      WRITE(6,1)
C      1 FORMAT(1H1,30HTABLE OF ORIGINAL ADJACENT NODE POINTS//
2 4X,4HNODE,13X,6HNO. OF,4X,6HNO. OF,27X,20HADJACENT NODE POINTS/
3 9X,1H4,9X,1H5,9X,1H9,9X,1H9,9X,1H7,9X,1H8//)
C      DO 2 I=1,MUNNP
C      2 WRITE(6,3) I,NAOJNP(I),NAOJEL(I),NPADJ(I,J),J=1,MXADJP)
C      3 FORMAT(1B,6X,21C,B110)
C
C      CALCULATE ORIGINAL BANDWIDTH
C
C      4 CONTINUE
C      MAXBO=0
C      DO 6 I=1,MUNNP
C        NUM=NAOJNP(I)
C        DO 6 J=1,NUM
C          NP=NPADJ(I,J)
C          NBAN=IABS(NP-I)
C          IF(NBAN.LE.MAXBO) GO TO 6
C          MAXBO=NBAN
C      6 CONTINUE
C
C      REMEMBER WITH PATH PROCEDURE
C
C      CALL PATH(MAXNP,MUNNP,MUMST,NSTART,NPT,NPTP,MXADJP,NAOJNP,NPAOJ,
11IGP,MUNGP)
C
C      NPTP = NEW NODE POINT NUMBERS IN OLD NODE POINT ORDER
C      NPTN = OLD NODE POINT NUMBERS IN NEW NODE POINT ORDER
C      PRINT
C
C      WRITE(6,8) MUNGP
C      8 FORMAT(1H1,24HNO. OF GRAND PARTITIONS=,15)
C      WRITE(6,9) ((IGP(I),I=1,MUNGP)
C      9 FORMAT(//23H PARTITION NEW NODE NO./12X,15,10X,(5))
C      450 WRITE(6,10) MAXBO
C      10 FORMAT(1H1,20H ORIGINAL BANDWIDTH=,15)
C
C      CALCULATE AND PRINT NEW BANDWIDTH
C
C      MAXBD=0
C      DO 11 I=1,MUNNP
C        NPNEW=NPTP(I)
C        NUM=NAOJNP(I)
C        DO 11 J=1,NUM
C          NP=NPADJ(I,J)
C          MPNEW=NPTP(NP)
C          NBAN=IABS(NPNEW-MPNEW)
C          IF(NBAN.LE.MAXBO) GO TO 11
C          MAXBO=NBAN
C      11 CONTINUE
C      WRITE(6,12) MAXBD
C      12 FORMAT(20H NEW BANDWIDTH=,15)
C
C      MIN(PIZE,BANDWIDTH WITH MINIMAX PROCEDURE AND CHECK
C      CALL MIN(I,MAXNP,MUNNP,NAOJNP,MXADJP,NPAOJ,NPTN,NPTP,S,
11MAXBO)
C      IF(MAXBO.LT.MXNP8) GO TO 14
C      WRITE(6,13)
C      13 FORMAT(//20H BANDWIDTH TOD LARGE)
C      CALL EXIT
C
C      REMEMBER NPAOJ
C
C      14 DO 15 I=1,MUNNP
C        KN=NACJNP(I)
C        DO 15 J=1,KN
C          KT=NPADJ(I,J)
C          NPADJ(I,J)=NPTP(KT)
C      15 NPADJ(I,J)=NPTP(KT)
C
C      PRINT
C
C      IF((PRINT.NE.1).AND.((PRINT.NE.8))) GO TO 19
C      WRITE(6,16)
C      16 FORMAT(1H1,30HTABLE OF NEW ADJACENT NODE POINTS//
14X,4HNEW,3X,4HOLD,6X,6HNO. OF,4X,6HNO. OF,27X,
22HADJACENT NODE POINTS/4X,4HNODE,3X,4HNODE,5X,9HADJ,PTS.,
31X,9HACJ,FLS.,5X,1H1,9X,1H2,9X,1H3,
4 9X,1H4,9X,1H5,9X,1H9,9X,1H7,9X,1H8//)
C      DO 17 I=1,MUNNP

```



```

      NMP(2,M)=NUNE
      NMP(3,M)=IZONE
      NMP(4,M)=KASE
      NMP(5,M)=NPI
      NMP(6,M)=NPJ
      NMP(7,M)=MPK
      NMP(8,M)=NPL
      WRITE(1,132) NUNE,IZONE,NPI,MPJ,MPK,NPL
132 FORMAT(16,11,3X,16,4X,16,4X,16,4X,16)
C
      IF(IPLAST(IZONE).EQ.0) GO TO 7
      NUNPL=NUNPL+1
      IF(NPL.NE.0) GO TO 133
      ITL=0
      TML=0.0
      RL=0.0
      ZL=0.0
      GO TO 134
133 ITL=ITYPE(NPL)
      TML=THETA(NPL)
      RL=R(NPL)
      ZL=Z(NPL)
134 WRITE(1,14) NUNE,IZONE,IPLAST(IZONE),NPI,MPJ,MPK,NPL,
      1 ITYPE(NPL),ITYPE(NPJ),ITYPE(MPK),ITL,
      2 THETA(NPL),THETA(NPJ),THETA(MPK),TML,
      3 R(NPL), R(NPJ), R(MPK),RL,
      4 Z(NPL), Z(NPJ), Z(MPK),ZL
      7 CONTINUE
C
      PEAD AND PRINT STARTING NODE DATA
C
      READ(5,100) ANUMF,MUNST
      READ(5,114) (INSTART(I),I=1,MUNST)
C
      MUNST=NO. OF STARTING NODES (LE=20)
      INSTART=STARTING NODE NUMBERS
C
      WRITE(1,135) MUNST
135 FORMAT(1H,10)STARTING NODE DATA//20H NO. OF START NODES=,15//)
      WRITE(1,136) (INSTART(I),I=1,MUNST)
136 FORMAT(22H STARTING NODE NUMBERS/(19X,15))
C
      PRINT 14
      RETURN
      END
C
      SUBROUTINE ORDER(NPI,MPJ,MPK,NPL,R,7,I,STRES,KASE,NANMP)
      IMPLICIT REAL*8(A-M,O-Z)
      DIMENSION R(NANMP),7(MANMP)
C
      ORDER NODE POINT LEFTPRING FOR ELEMENT AND OFFINE CASE
C
      R = RADIAL COORDINATE OF NODE POINT
      Z = VERTICAL COORDINATE OF NODE POINT
      KASE = 1 GENERAL TRIANGLE

```

```

      = 2 TRIANGLE, ONE NODE ON Z-AXIS
      = 3 TRIANGLE, TWO NODES ON Z-AXIS
      = 4 GENERAL RECTANGLE
      = 5 RECTANGLE, ONE NODE ON Z-AXIS
      = 6 RECTANGLE, TWO NODES ON Z-AXIS
      ISTRS = 0 AXISYMMETRIC PROBLEM
      = 1 PLANE STRAIN PROBLEM
      = 2 PLANE STRESS PROBLEM

```

```

      NPI=MPJ
      NPJ=MPK
      MPK=MPK
      NPL=IPL
      IF(IPLAST(IZONE).LT.R(NPJ)) GO TO 1
      IF(IPLAST(IZONE).LT.R(NPJ)) GO TO 2
      IF(IPLAST(IZONE).LT.Z(NPJ)) GO TO 1
2
      NPI=MPJ
      NPJ=MPK
      MPK=MPK
      NPL=IPL

```

```

1
      IF(IPLAST(IZONE).LT.R(NPJ)) GO TO 3
      IF(IPLAST(IZONE).LT.R(NPJ)) GO TO 4
      IF(IPLAST(IZONE).LT.Z(NPJ)) GO TO 3
4
      NPI=MPK
      NPJ=MPK
      MPK=MPK
      NPL=IPL

```

```

3
      IF(IPLAST(IZONE).LT.R(NPJ)) GO TO 5
      IF(IPLAST(IZONE).LT.R(NPJ)) GO TO 5
      IF(IPLAST(IZONE).LT.Z(NPJ)) GO TO 6
      IF(IPLAST(IZONE).LT.Z(NPJ)) GO TO 5
6
      NPI=MPK
      NPJ=MPK
      MPK=MPK
      NPL=IPL

```

```

      NODE POINT 1 IN PROPER LOCATION (CLOSEST TO ORIGIN)

```

```

5
      AJ=R(NPJ)-R(NPI)
      BJ=Z(NPJ)-Z(NPI)
      AK=R(NPJ)-R(NPI)
      BK=Z(NPJ)-Z(NPI)
      HJK=AJ*BK-AK*BJ
      IF(MPL.NE.0) GO TO 7

```

```

      TRIANGULAR ELEMENT

```

```

      IF(HJK.GT.0.) GO TO 8
      MPK=NPJ
      NPJ=MPK
      8 IF(IPLAST(IZONE).EQ.0.) GO TO 9
      10 KASE=1
      RETURN
      9 IF(ISTRS.NE.0) GO TO 10
      IF(IPLAST(IZONE).EQ.0.) GO TO 11
      KASE=2

```



```

C      IF(MAXBOP.LE.MAXBO) GO TO 12
C
C      DO 5 I=1,NUMNP
C      NPOLD=NPTN(I)
C      5 NPTN(NPOLD)=I
C      MAXBCP=MAXBO
C      GO TO 3
C
C      12 DO 16 I=1,NUMNP
C      NPNEW=NPTN(I)
C      16 NPTN(NPNEW)=I
C      RETURN
C      END
C
C      SUBROUTINE GSDRTS(IARRAY,NRCDS,NWRDS,NKEY,IOUTAP)
C      IMPLICIT REAL*8(A-H,O-Z)
C      DIMENSION IARRAY(NWRDS,NRCDS)
C
C      SORT DATA AND WRITE ONTO TAPE
C
C      REMIND IOUTAP
C      CALL SORT1(IARRAY,IARRAY,NRCDS,NWRDS,NKEY,C)
C      DO 3 I=1,NRCDS
C      3 WRITE(IOUTAP)(IARRAY(J,I),J=1,NWRDS)
C      REMIND IOUTAP
C      RETURN
C      END
C
C      SUBROUTINE SORT1(IARRAY,JARRAY,NRCDS,IWRDS,JWRDS,IFFY,ISWT)
C      IMPLICIT REAL*8(A-H,O-Z)
C      DIMENSION IARRAY(IWRDS,NRCDS),JARRAY(JWRDS,NRCDS)
C
C      GENERAL PURPOSE SORTER
C
C      LOGICAL CHECK
C      M=NRCDS-1
C
C      1 CHECK=.FALSE.
C      DO 6 I=1,2
C      DO 2 J=I,M,2
C      IF(IARRAY(IKEY,J).LE.IARRAY(IKEY,J+1)) GO TO 2
C      2 J=I+1,IWRDS
C      ITEMP=IARRAY(K,J)
C      IARRAY(K,J)=IARRAY(K,J+1)
C      IARRAY(K,J+1)=ITEMP
C      3 IF(ISWT.EQ.O) GO TO 5
C      DO 4 K=I,JWRDS
C      JTEMP=JARRAY(K,J)
C      JARRAY(K,J)=JARRAY(K,J+1)
C      JARRAY(K,J+1)=JTEMP
C      4 JARRAY(K,J+1)=JTEMP
C      5 CHECK=.TRUE.
C      2 CONTINUE
C      6 CONTINUE

```

```

C
C
C
2 IF(MXNMP.LT.MXNPB) NUMNPB=NUMNP
  IF(NUMNP.GE.MXNPB) NUMNPR=MXNPB
  DO 3 I=1,NUMNPB
    READ(8) NPN,NADJNP(I),NADJEL(I),(NPAJ(I,J),J=1,MXADJJP)
    READ(4) NPN,R(I),Z(I),ITYPE(I),THETA(I)
  3 CONTINUE
C
C
C
READ ANOTHER ELEMENT DATA RECORD
ICOUNT=0
4 READ(1) KEY,NUME,IZONE,KASF,NTL,NTJ,NTK,NTL
  ICOUNT=ICOUNT+1
  LNP=MAXD(NTI,NTJ,NTK,NTL)
C
C
C
CHECK TO SEE IF LARGEST NODE POINT IN THIS ELEMENT FALLS OUTSIDE
  BUFFER
C
C
C
IF(LNP=NPOUT).GT.MXNPB) GO TO 100
C
C
C
PROCESS THIS ELEMENT
C
C
C
6 NP=(NTI-NPOUT)
  NPJ=NTJ-NPOUT
  NPK=NTK-NPOUT
  IF(NTL.EQ.0) NPL=D
  IF(NTL.NE.0) NPL=NTL-NPOUT
  S1=0.0
  C1=0.0
  IE=1,ELAST(IZONE)
  A1=E1(IZONE)
  A2=E2(IZONE)
  A3=E3(IZONE)
  A4=E4(IZONE)
  A5=E5(IZONE)
  RHO=WT(IZONE)/(386.4*1728.)
  CALL ELAST(IE,ISTRES,A1,A2,A3,A4,A5,C,NUME)
  CALL STIFF(KASE,NPI,NPJ,NPK,NPL,NUME,MXNPB,ISTRES,C,R,2,CK,AINT,
    I51,C1)
  CALL ADJUST(MXNPB,CK,ITYPE,THETA,NADJEL,NPI,NPJ,NPK,NPL)
  CALL DISTK(MXNPB,MXADJP,CK,SNPUU,SNPUM,SNPVM,SAOUU,SAOUM,SAOWU,
    ISADWM,NPI,NPJ,NPK,NPL,NPAJ,NPOUT)
  CALL MASS(MXNPB,RHO,R,2,AINT,XMASS,S1,C1,NPI,NPJ,NPK,NPL,ISTRES)
  WRITE(12) KEY,NUME,IZONE,NTI,NTJ,NTK,NTL,((C(I,J),I=1,4),J=1,4),
    I,KASE,S1,C1
C
C
C
GO TO READ ANOTHER ELEMENT RECORD OR SET STOP SWITCH
C
C
C
IF(ICOUNT.LT.NUMEL) GO TO 4
  KEND=1
  KEY=NUMNP+1
C
C
C
SET-UP FOR DUMPING BUFFERS
C
C
C
100 NUMCP=KEY-1
  NUMNPB=NUMCP-NPOUT

```

25

```

C
C
C
PRINT AND WRITE FINISHED NODE DATA ON TAPE
C
C
C
113 CALL PRNK(MXNPB,MXADJP,NADJNP,NPAJ,NADJEL,I,PRINT,SNPUU,
  1SNPUM,SNPVM,SAOUU,SAOUM,SAOWU,THETA,ITYPE,XMASS,NPOUT,
  2NUMNPB)
  DO 101 I=1,NUMNPR
    101 WRITE(3) I,NADJNP(I),ITYPE(I),THETA(I),XMASS(I),SNPUU(I),
      1SNPUM(I),SNPVM(I),NPAJ(I,J),SAOUU(I,J),SAOWU(I,J),
      2SAOWU(I,J),J=1,MXADJP)
C
C
C
UPDATE PERIOD DATA
C
C
C
104 DO 105 I=1,NUMNPR
  IF(ITYPE(I).EQ.2) GO TO 106
  OUM=SNPUU(I)/XMASS(I)
  IF(DUM.LT.OUM) GO TO 106
  OUM=DUM
  105 NPOUT=I
  IF(ITYPE(I).EQ.2.OR.ITYPE(I).EQ.1) GO TO 105
  OUM=SNPVM(I)/XMASS(I)
  IF(DUM.LT.OUM) GO TO 105
  OUM=DUM
  105 NPOUT=I
  105 CONTINUE
C
C
C
SKIP OUT FOR LAST ELEMENT COMPLETED OR GO TO REFILL NODE ARRAYS
C
C
C
IF(KEND.EQ.1) GO TO 300
  NPR=MXNPB-MXNMPB
  GO TO 902
C
C
C
GO TO ERASE REMAINDER OF BUFFERS
C
C
C
107 KX=NPR+1
  ISWCH=2
  GO TO 900
C
C
C
FILL UP REMAINDER OF ARRAYS WITH NEW NODE DATA
C
C
C
108 IF((NUMNP-NUMCP).LT.MXNPB) KNP=NUMNP-NUMCP-NPR
  IF((NUMNP-NUMCP).GE.MXNPB) KNP=MXNPB-NPR
  DO 109 I=1,KNP
    L=NPR+I
    READ(8) NPN,NADJNP(L),NADJEL(L),(NPAJ(L,J),J=1,MXADJP)
    READ(4) NPN,R(L),Z(L),ITYPE(L),THETA(L)
  109 CONTINUE
  NPOUT=NUMCP
  GO TO 6
C
C
C
MOVE UNPROCESSED DATA TO TOP OF BUFFERS
C
C
C
902 DO 903 K=1,NPR
  L=NUMNPB+K
  NADJNP(K)=NADJNP(L)
  NADJEL(K)=NADJEL(L)

```



```

C(2,2)=C(1,1)
C(2,3)=C(1,3)
C(3,1)=C(1,3)
C(3,2)=C(2,3)
C(3,3)=E2
C(4,4)=E4
RETURN
C
ANISOTROPIC PLANE STRESS PROBLFM
C
C
2 C(1,1)=2.*E5*(E1-2.*E5)/E1
C(1,3)=2.*E3*E5/E1
C(3,1)=C(1,3)
C(3,3)=E2-E3*E2/E1
C(4,4)=E4
RETURN
C
21 WRITE(6,3) IELAST,NUME,ISTRES
3 FORMAT(1H1/31H ERROR IN ELASTIC CONSTANT DATA/
113H IELAST =,15/13H ELEMENT NO.=,15/,
213H ISTRES
=,15)
CALL EXIT
C
COMPRESSIBLE FLUID
C
30 IF(IELAST.NE.3) GO TO 21
C
IF(ISTRES.EQ.2) GO TO 21
C
DO 31 I=1,3
DO 31 J=1,3
31 C(I,J)=E1
C
RETURN
ENC
C
SUBROUTINE STIFF(KASE,NPI,NPJ,NPK,NPL,NUME,MAXNP,ISTRES,C,R,Z,
1CK,AT,SI,CI)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION C(4,4),R(MAXNP),Z(MAXNP),CK(8,8),A(23),D(8,8),G(8,8),
1VEC(8)
C
COMPUTE ELEMENT STIFFNESS MATRIX
C
KASE = 1 GENERAL TRIANGLE
= 2 NODE I ON Z-AXIS
= 3 NODES I, K ON Z-AXIS
= 4 GENERAL RECTANGLE
= 5 NODE I ON Z-AXIS
= 6 NODES I, L ON Z-AXIS
C
= ELASTIC MODULI MATRIX
CK = STIFFNESS MATRIX
AT = INTEGRALS FOR COMPUTING K AND M
C
DO 1 I=1,8

```

```

A=DSQRT(AJ*AJ+RJ*RJ)
AL=R(NPL)-R(NPT)
8L=Z(NPL)-Z(NPT)
8=DSQRT(AL*AL+8L*8L)
M=A*8

```

```

IF(KASE.NE.4) GO TO 6

```

```

C(1,1)=1.0

```

```

O(2,1)=-B/H

```

```

O(3,1)=1/H

```

```

O(4,1)=-A/H

```

```

O(5,2)=1.0

```

```

O(6,2)=-8/H

```

```

O(7,2)=1/H

```

```

O(8,2)=-A/H

```

```

O(2,3)=-C(6,2)

```

```

O(3,3)=-C(7,2)

```

```

O(6,4)=C(2,3)

```

```

O(7,4)=D(3,3)

```

```

O(3,5)=O(7,2)

```

```

O(7,6)=O(7,2)

```

```

IF(KASE.EQ.6) GO TO 7

```

```

O(3,7)=D(3,3)

```

```

O(4,7)=-O(8,2)

```

```

O(7,8)=O(7,4)

```

```

O(8,8)=-D(8,2)

```

```

NORD=8

```

```

IF(KASE.NE.4) GO TO 8

```

```

IF(ISTRES.NE.0) GO TO 8

```

```

G(1,1)=C(2,2)*AI(5)

```

```

G(2,1)=C(1,2)*C(1,1)*AI(1)+C(2,2)*AI(7)

```

```

G(3,1)=C(1,2)*C(1,1)*AI(2)+S(1,1)*AI(3)+C(2,2)*AI(9)

```

```

G(4,1)=C(1,2)*S(1,1)*AI(1)+C(2,2)*AI(6)

```

```

G(6,1)=-C(2,3)*S(1,1)*AI(1)

```

```

G(7,1)=C(2,3)*C(1,1)*AI(3)-S(1,1)*AI(2)

```

```

G(6,1)=C(2,3)*C(1,1)*AI(1)

```

```

DUM1=C(1,1)*AI(14)+S(1,1)*AI(13)

```

```

DUM2=C(1,1)*AI(13)-S(1,1)*AI(14)

```

```

DUM3=C(1,1)*AI(12)+2.*S(1,1)*AI(15)+S(1,1)*AI(11)

```

```

DUM4=S(1,1)*AI(12)-2.*S(1,1)*AI(15)+C(1,1)*AI(11)

```

```

G(2,2)=C(1,1)*C(1,1)*AI(4)+2.*C(1,2)*AI(3)+C(2,2)*AI(10)

```

```

1+C(4,4)*S(1,1)*AI(4)

```

```

G(3,2)=C(1,1)*C(1,1)*DUM1+C(1,2)*AI(2)+C(1,1)*AI(16)+S(1,1)*AI(18)

```

```

1+C(2,2)*AI(17)-C(4,4)*S(1,1)*DUM2

```

```

G(7,2)=C(1,3)*C(1,1)*DUM2+C(2,3)*C(1,1)*AI(10)-S(1,1)*AI(16)

```

```

1-C(4,4)*S(1,1)*DUM1

```

```

G(8,2)=AI(4)*C(1,3)*C(1,1)*C(4,4)*S(1,1)*C(2,3)*C(1,1)*AI(3)

```

```

G(3,3)=C(1,1)*DUM3+2.*C(1,2)*C(1,1)*AI(20)+S(1,1)*AI(23)

```

```

1+C(2,2)*AI(21)+C(4,4)*DUM4

```

```

G(6,3)=-C(1,3)*S(1,1)*DUM1-C(2,3)*S(1,1)*AI(16)+C(4,4)*C(1,1)*DUM2

```

```

G(7,3)=C(1,3)*C(4,4)*S(1,1)*C(1,1)*AI(11)-AI(12)+AI(15)+C(1,1)*S(1,1)

```

```

1+C(2,3)*C(1,1)*AI(23)-S(1,1)*AI(20)

```

```

G(8,3)=C(1,3)*C(1,1)*DUM1+C(2,3)*C(1,1)*AI(16)+C(4,4)*S(1,1)*DUM2

```

```

G(6,6)=C(3,3)*S(1,1)*C(4,4)*C(1,1)*AI(4)

```

```

G(7,6)=-C(3,3)*S(1,1)*DUM2+C(4,4)*C(1,1)*DUM1

```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```


```

```

A=AJ-AK
R1=R1NPJ1(
C
1F(ISTRES.EQ.0.1 GO TO 2
A1(4)=M/2.
A1(13)=H*(AJ+AK)/6.
A1(14)=H*(BJ+AK)/6.
RETURN
C
2 A1(1)=M/2.
A1(2)=H*(BJ+AK)/6.
A1(3)=H*(AJ+AK)/6.
A1(4)=R1*A1(1)+A1(3)
1F(KASE.EQ.1 GO TO 3
A1(9)=A1(2)
A1(10)=A1(4)
C
3 1COUNT=1
RA=R1
R2=R1NPJ1
C=BJ/AJ
O=O.
DUM=-1.
1F(C.EQ.0.1 GO TO 100
1F(KASE.EQ.1 GO TO 102
1F(KASE.NE.2 GO TO 104
1F(KASE.NE.0.1 GO TO 102
FC=OLOG(RR)
GO TO 104
102 FC=OLOG(RR/RA)
104 DUM1=RB-RA
DUM2=RB*RB-RA*RA
DUM3=RB*RB*RB-RA*RA*RA
DUM4=DUM2*(RB*RR+RA*RA)
1F(KASE.EQ.3 GO TO 103
F1=DUM1-R1*FO
F2=DUM2/2.-2.*R1*(DUM1+R1*FO)
F3=DUM3/3.-1.5*R1*DUM2+3.*R1*(DUM1-R1*FO)
G0=DUM2/2.
G1=DUM3/3.-R1*(DUM2/2.
G2=DUM4/4.-2.*R1*DUM3/3.*R1*(DUM2/2.
C
1F(KASE.NE.1 GO TO 105
A1(5)=A1(1)+DUM1*(C*F1+D*FN)
A1(6)=A1(1)+DUM1*(C*F2/2.+C*D*F1+D*FN/2.
A1(7)=A1(1)+DUM1*(C*F2+D*F1)
A1(9)=A1(1)+DUM1*(C*F3/2.+C*D*F2+D*F1/2.
A1(10)=A1(1)+DUM1*(C*F3+D*F2)
GO TO 106
105 1F(KASE.NE.2 GO TO 107
106 A1(8)=A1(1)+DUM1*(C*F3/3.+C*D*F2+C*FN/3.+D*FN/3.)
107 A1(13)=A1(1)+DUM1*(C*G2+D*G1)
A1(14)=A1(1)+DUM1*(C*G2/2.+C*D*G1+D*G2/2.)
C
100 GO TO(201,202,203), 1COUNT
201 1COUNT=2
1F(A.EQ.0.1 GO TO 100
RB=R1NPJ1
RA=R1NPJ1
C=BJ/A
O=H/A
DUM=-1.
GO TO 101
202 1COUNT=3
1F(KASE.EQ.0.1 GO TO 100
1F(KASE.EQ.0.1 GO TO 100
RB=R1NPJ1
RA=R1
C=BJ/AK
O=O.
DUM=-1.
GO TO 101
203 RETURN
C
RECTANGULAR ELEMENT
C
300 AJ=R1NPJ1-R1NPJ1
BJ=Z1NPJ1-Z1NPJ1
A=OSQRT(AJ+BJ*BJ)
AL=R1NPJ1-R1NPJ1
BL=Z1NPJ1-Z1NPJ1
R=OSQRT(AL+BL*BL)
H=A*H
S1=-BJ/A
C1=AJ/A
R1=R1NPJ1
1F(S1.GT..01 GO TO 301
S1=0.
C1=1.
GO TO 302
301 1F(C1.GT..01 GO TO 302
S1=1.
C1=0.
S2=S1*S1
S3=S2*S1
C2=C1*C1
C3=C2*C1
C
1F(ISTRES.EQ.0 GO TO 303
A1(4)=H
A1(11)=A*H/3.
A1(12)=B*H/3.
A1(13)=A*H/2.
A1(14)=B*H/2.
A1(15)=H*H/4.
RETURN
C
303 A1(1)=H
A1(2)=B*H/2.
A1(3)=A*H/2.
A1(4)=R1*(A1(1)+C1*(A1(3)+S1*A1(2)
A1(8)=A*H/3.

```



```

MX=3
LX=1
LY=5
LZ=7
GO TO 100
C
4 NI=NPK
NJ=NPI
NK=NPJ
NL=NPL
MX=5
LX=1
LY=3
LZ=7
GO TO 100
C
5 IF(NPL.EQ.0) GO TO 6
NI=NPL
NJ=NPI
NK=NPJ
NL=NPK
MX=7
LX=1
LY=3
LZ=5
GO TO 100
C
6 RETURN
C
100 SNUUINI)=SNUU(11)*RK(MX,MX)
SNUUINI)=SNUUINI)+RK(MX,MX*1)
SNPWINI)=SNPWINI)+RK(MX*1,MX*1)
GO 101 1=1,MXADJP
J=1
IF(INPADJINI,11-NPOUT),FQ,NJ) GO TO 102
101 CCNTIME
205 NPI=NPI+NPOUT
NPJ=NPJ+NPOUT
NPK=NPK+NPOUT
IF(NPL.EQ.0) GO TO 204
NPL=NPL+NPOUT
204 IF(1E16,203) NPI,NPJ,NPK,NPL,NI,NJ,NK,NL,NPOUT,
1(NPADJINI,11,1=1,MXADJP)
203 FORMAT(1M/32M ERROR IN STIFFNESS DISTRIBUTION//
1 13M NPI =.15/13M NPJ =.15/13M NPK
213M NPL =.15/10X,515/10X,B15)
CALL EXIT
C
102 SADDUINI,J)=SADDUINI,J)+RK(MX,LX)
SADDUINI,J)=SADDUINI,J)+RK(MX,LX*1)
SADDUINI,J)=SADDUINI,J)+RK(MX*1,LX)
SADDUINI,J)=SADDUINI,J)+RK(MX*1,LX*1)
C
103 SADDUINI,J)=SADDUINI,J)+RK(MX,LX)
SADDUINI,J)=SADDUINI,J)+RK(MX,LX*1)
SADDUINI,J)=SADDUINI,J)+RK(MX*1,LX)
SADDUINI,J)=SADDUINI,J)+RK(MX*1,LX*1)
C
104 SADDUINI,J)=SADDUINI,J)+RK(MX,LX)
SADDUINI,J)=SADDUINI,J)+RK(MX,LX*1)
SADDUINI,J)=SADDUINI,J)+RK(MX*1,LX)
SADDUINI,J)=SADDUINI,J)+RK(MX*1,LX*1)
C
105 IF(NL.EQ.0) GO TO 105
GO 106 1=1,MXADJP
J=1
IF(INPADJINI,11-NPOUT),FQ,NL) GO TO 107
106 CONTINUE
GO TO 205
107 SADDUINI,J)=SADDUINI,J)+RK(MX,LZ)
SADDUINI,J)=SADDUINI,J)+RK(MX,LZ*1)
SADDUINI,J)=SADDUINI,J)+RK(MX*1,LZ)
SADDUINI,J)=SADDUINI,J)+RK(MX*1,LZ*1)
GO TO 1
C
2NO
C
SUBROUTINE ADJUST(MAXNP,BK,ITYPE,THETA,NAOJEL,NPI,NPJ,NPK,NPL)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION RK(8,8),ITYPE(MAXNP),THETA(MAXNP),NAOJEL(MAXNP),NODE(4)
C
C DO ROLLER AND PINNED NODE MODIFICATIONS
LIM=4
IF(NPL.EQ.0) LIM=3
NODE(1)=NPI
NODE(2)=NPJ
NODE(3)=NPK
NODE(4)=NPL
GO 100 1=1,LIM
IN=NODE(1)
IND=2*1-1
SFAC=1.0/NAOJEL(IN)
C
IF(ITYPE(IN),NE,2) GO TO 10
BK(INO,INO)=SFAC
BK(INO*1,INO)=0.0
RK(INC,INC*1)=0.0
BK(INC*1,IND*1)=SFAC
GO TO 20
C
10 IF(ITYPE(IN),NF,1) GO TO 20
S=DSIN(THETA(IN))
C=DCOS(THETA(IN))
RK(INO,INO)=RK(INO,(NO,IND*1)+C=S*RK(INO*1,IND*1)
1.0*S
RK(INC*1,IND)=0.0

```

```

C
      RK(IND,IND+1)=0.0
      RK(IND+1,IND+1)=SFAC
C
      20 DO ICD J=1,LIM
      IF(1.EQ.J) GO TO 100
      JN=MODE(J)
      JNC=2*J-1
C
      IF(.NOT.(ITYPE(IN).EQ.2.OR.ITYPE(JN).EQ.2)) GO TO 50
C
      RK(IND,JND)=0.0
      RK(IND+1,JND)=0.0
      RK(IND,JND+1)=0.0
      RK(IND+1,JND+1)=0.0
      GO TO 100
C
      50 IF(ITYPE(IN).NE.1) GO TO 75
C
      S=DSIN(THETA(IN))
      C=DCOS(THETA(IN))
      RK(IND,JND)=RK(IND,JND)+C*RK(IND+1,JND)+S
      RK(IND,JND+1)=RK(IND,JND+1)+C*RK(IND+1,JND+1)+S
      RK(IND+1,JND)=0.0
      RK(IND+1,JND+1)=0.0
C
      75 IF(ITYPE(JN).NE.1) GO TO 100
C
      S=DSIN(THETA(JN))
      C=DCOS(THETA(JN))
      RK(IND,JND)=RK(IND,JND)+C*RK(IND,JND+1)+S
      RK(IND+1,JND)=RK(IND+1,JND)+C*RK(IND+1,JND+1)+S
      RK(IND,JND+1)=0.0
      RK(IND+1,JND+1)=0.0
C
      100 CONTINUE
C
      RETURN
      ENI
C
      SUBROUTINE MASS(MAXNP,RMD,R,Z,AL,XMASS,SI,CI,NPI,NPJ,NPK,NPL,
      11STRESS)
      IMPLICIT REAL(A-H,O-Z)
      DIMENSION R(MAXNP),Z(MAXNP),XMASS(MAXNP),AL(23)
C
      COMPUTE AND DISTRIBUTE MASS TO NODE POINTS
C
      IF(NPL.NE.0) GO TO 2
C
      AJ=R(NPJ)-R(NCI)
      AK=R(NPK)-P(NPI)
      AL=Z(NPJ)-Z(NPI)
      BL=Z(NPK)-Z(NPI)
      M=AJ*BL-AL*AK
      B=DJ-PK
      A=AJ-AK
      DATA AI/6HBEFORE,6HAFTR,6HSTIFFN,6HMESS,6HTARLFS,
      16HBCINDA,6HRY ALT,6HEPATIO,6HN,6H NOME,6HNPJ,
      26HSPM,6HNDJACE,6HNT,6HSDIU,6HSDUM,6HSDUM,
      36HSTRESS,6HSTRNDU,6HSTRNDU,6HSTNDU,6HSTNDU,6HSTNDU,
      46HSTNDU,6HSTNDU,6HSTRNDU,6HSTRNDU,6HSTNDU,6HSTNDU,
      56HSTZADM,6HSTZADM,6HSTZADM,6HSTZADM,6HSTZADM,6HSTZADM,
      /

```

```

C      GO TO 25
14 WRITE(6,112) A(13),A(14),A(15),A(16),A(17),A(18),A(19),A(20),
   1 A(21),K=6,9)
112 FORMAT(1M,12A6)
113 WRITE(6,113) A(10),A(11),A(12),A(13),A(14),A(15),A(16),A(17),A(18),A(19),A(20),
   1 A(21),K=11,13)
113 FORMAT(17A6,81M,1A6)
GO 15 1=1,NUMP
K=1,NPOT
15 WRITE(6,114) K,SNPUM(1),SNPUM(1),SNPUM(1)
114 FORMAT(15,3X,10F14.4)
C
      ICOUNT=1
      JCOUNT=16
16 WRITE(6,112) A(11),A(12),A(13),A(14),A(15),A(16),A(17),A(18),A(19),A(20),
   1 A(21),K=11,13)
112 FORMAT(1M,12A6)
GO TO 17,19,21,23,ICOUNT
17 DO 18 1=1,NUMP
   1 NUM=NADJNP(1)
   1 K=1,NPOT
18 WRITE(6,114) K,(SADUM(I,J),J=1,NUM)
   1 ICOUNT=2
   1 JCOUNT=17
   1 GO TO 16
19 DO 20 1=1,NUMP
   1 NUM=NADJNP(1)
   1 K=1,NPOT
20 WRITE(6,114) K,(SADUM(I,J),J=1,NUM)
   1 ICOUNT=3
   1 JCOUNT=18
   1 GO TO 16
21 DO 22 1=1,NUMP
   1 NUM=NADJNP(1)
   1 K=1,NPOT
22 WRITE(6,114) K,(SADUM(I,J),J=1,NUM)
   1 ICOUNT=4
   1 JCOUNT=19
   1 GO TO 16
23 DO 24 1=1,NUMP
   1 NUM=NADJNP(1)
   1 K=1,NPOT
24 WRITE(6,114) K,(SADUM(I,J),J=1,NUM)
   1 ICOUNT=5
   1 JCOUNT=20
   1 GO TO 16
25 ISTEIFF=2
C
30 IF(IPRINT.EQ.3) GO TO 14
   1 IF(IPRINT.EQ.8) GO TO 14
C
56 IF(IPRINT.EQ.6) GO TO 58
   1 IF(IPRINT.EQ.8) GO TO 58
   1 GO TO 60
C
59 WRITE(6,112) A(13),A(14),A(15),A(16),A(17),A(18),A(19),A(20),
   1 A(21),K=11,13)
112 FORMAT(1M,12A6)

```

```

DO 59 1=1,NUMP,E
L=1,NPOT
NUM=1+7
IF(NUM.GT.NUMP) NUM=NUMP
59 WRITE(6,114) L,(XMASS(J),J=1,NUM)
60 RETURN
END
C
SUBROUTINE LINKIC
IMPLICIT REAL*8(A-H,O-Z)
COMMON MAXNP,MAXCJP,MXLINE,MXLOAD,MXMPA,MAXELA,MXELA,NUMP,MUPEL,
   1 ISTRS,LINES,MUPEL,LOADNP(1),PRTD(1),PRTD(1),PRTD(1),PRTD(1),
   2 RAD(1,50),ZAD(1,50),SNOW(1,50),TITLE(1),MAXRD,
   3 MLOCK,NREADS,MAXWK,TREAL,POAMP,BETA,KRUM,
   4 MZONE,MZONES,IPRINT,MPTN,IELAST,MGT,E1,E2,E3,E4,E5
   1 DIMENSION MPTN(1000),IELAST(20),MGT(20),E1(20),E2(20),E3(20),E4(20),
   1 E5(20)
   1 DIMENSION C(4,4)
   1 DIMENSION NADJNP(100),NPADJ(100),NADJEL(100)
   1 DIMENSION R(100),Z(100),ITYPE(100),THETA(100)
   1 DIMENSION STMPU(4,100),STMPW(4,100),STADU(4,100),STADW(4,100),R
C
THIS LINK COMPUTES ELEMENT STRESSES, MODIFIES THEM FOR ROLLER
SUPPORTS, AND DISTRIBUTES THEM TO NODE POINT STRESSES
C
FLOW OF THIS LINK IS SAME AS LINKR
C
REWIND 4
REWIND 8
REWIND 12
C
KEND=0
NPOT=0
NUMCP=0
NUMPB=0
NPR=MAXNP
KX=1
1 ISTEICH=1
GO TO 900
C
2 IF(NUMP.LT.MXNP) NUMP=NUMP
   1 IF(NUMP.GE.MXNP) NUMP=MAXNP
   1 DO 3 1=1,NUMP
   1 READ(8) NPN,NADJNP(1),NADJEL(1),NADJEL(1),NADJEL(1),NADJEL(1),
   1 READ(4) NPN,R(1),Z(1),ITYPE(1),THETA(1)
   1 CONTINUE
C
ICOUNT=0
4 READ(12) KEY,MUPEL,MZONES,MPTN,MTJ,MTR,MTL,(IC(1,1),I=1,4),J=1,4),
   1 KASE,S1,C1
   1 ICOUNT=ICOUNT+1
   1 LAP=MAX(MPTN,MTJ,MTR,MTL)
   1 IF(LAP.NPOT).GT.MXNP GO TO 100

```



```

C
C
C
1F(NPL.NE.0) GO TO 200
TRIANGULAR ELEMENT
AJ=RA(NPJ)-RA(NPI)
AK=RA(NPK)-RA(NPI)
BJ=ZA(NPJ)-ZA(NPI)
BK=ZA(NPK)-ZA(NPI)
M=AJ*BK-AK*BJ
N=BJ-AK
A-AJ-AK
25 IF(KASE.NE.1) GO TO 26
IF(ISTRES.NE.0) GO TO 27
ADR=1./RR
ROR=R/RR
ZOR=Z/RR
GO TO 28
27 IF(ICOUNT.NE.1) GO TO 100
ADR=0.0
ROR=0.0
ZOR=0.0
28 DUM1=B/M
DUM2=ADR*(RORR-AOZNR)/M
OC 35 I=1,3
35 S(1,1)=C(1,1)*DUM1+C(1,2)*DUM2
S(4,1)=C(4,4)*A/M
GO TO 29
26 ROR=1.0
IF(KASF.NE.2) GO TO 30
IF(ICOUNT.NE.1) GO TO 31
ZOR=0.0
GO TO 29
31 ZOR=Z/RR
GO TO 29
30 ZOR=0.0
IF(ICOUNT.NE.1) GO TO 100
29 OC 36 I=1,3
36 S(1,2)=C(1,3)*A/M
S(4,2)=C(4,4)*B/M
DUM1=BK/M
DUM2=(BK*ROR-AK*ZNR)/M
GO 37 I=1,3
37 S(1,3)=C(1,1)*DUM1+C(1,2)*DUM2
DUM1=AK/M
S(4,3)=C(4,4)*DUM1
GO 38 I=1,3
38 S(1,4)=C(1,3)*DUM1
S(4,4)=C(4,4)*B/M
DUM1=AJ/M
OC 39 I=1,3
39 S(1,6)=C(1,3)*DUM1
S(4,6)=C(4,4)*B/M
IF(KASF.EQ.3) GO TO 30
DUM1=-R/M
DUM2=(AJ*ZNR-AJ*BNR)/M

```

```

OC 40 I=1,3
40 S(1,5)=C(1,1)*DUM1+C(1,2)*DUM2
S(4,5)=C(4,4)*AJ/M
GO TO 300
C
C
C
RECTANGULAR ELEMENT
200 AJ=RA(NPJ)-RA(NPI)
BJ=ZA(NPJ)-ZA(NPI)
A=OSQRT(AJ*AJ+BJ*BJ)
AL=RA(NPL)-RA(NPI)
BL=ZA(NPL)-ZA(NPI)
O=OSQRT(AL*AL+BL*BL)
M=AOA
C
50 IF(KASE.NE.4) GO TO 51
IF(ISTRES.NE.0) GO TO 52
ADR=1./RR
ROR=R/RR
ZOR=Z/RR
GO TO 53
52 ADR=0.0
ROR=0.0
ZOR=0.0
53 DUM1=(RPOSL*ZPOCL-AOSI-BOCI)/M
DUM2=ADR*(ROR*ZP-RORR-AOZNR)/M
OC 45 I=1,3
65 S(1,1)=C(1,1)*DUM1+C(1,2)*DUM2
S(4,1)=C(4,4)*(BOCI-AOCI+RPOCI-ZPOSI)/M
GO TO 54
51 IF(KASE.EQ.5) GO TO 73
ROR=1.
ZOR=0.
GO TO 54
73 IF(ICOUNT.NE.1) GO TO 74
ROR=C1
ZOR=1
GO TO 54
74 ROR=R/RR
ZOR=Z/RR
54 DUM1=(RPSI-AOCI+RPOCI-ZPOSI)/M
OC 66 I=1,3
66 S(1,2)=C(1,3)*DUM1
S(4,2)=C(4,4)*(RPOSI+ZPOCI-AOCI-AOSI)/M
DUM1=(RPSI-RPOCI-ZPOCI)/M
DUM2=(RORR-ROR*ZP)/M
GO 67 I=1,3
67 S(1,3)=C(1,1)*DUM1+C(1,2)*DUM2
DUM1=(RPSI-RPOCI+ZPOSI)/M
S(4,3)=C(4,4)*DUM1
OC 68 I=1,3
68 S(1,4)=C(1,3)*DUM1
S(4,4)=C(4,4)*(RPSI-RPOCI-ZPOCI)/M
DUM1=(RPSI+ZPOCI)/M
DUM2=ROR*ZP/M
OC 69 I=1,3

```

```

69 S(1,5)=C(1,1)*DUM1+C(1,2)*DUM2
   DUM1=(RPOCL-ZPOSL)/H
   S(4,5)=C(4,4)*DUM1
   DO 70 I=1,3
70 S(1,6)=C(1,3)*DUM1
   S(4,6)=C(4,4)*(ZPOCL+RPOSL)/H
   DUM1=(A0CL-RPOCL+ZPOSL)/H
   DO 71 I=1,3
71 S(1,8)=C(1,3)*DUM1
   S(4,8)=C(4,4)*(A0S1-RPOSL-ZPOCL)/H
   IF(KASE.EQ.6) GO TO 300
   S(4,7)=C(4,4)*DUM1
   DUM1=(A0S1-RPOSL-ZPOCL)/H
   DUM2=(A0ZOR-RPOZP)/H
   DO 72 I=1,3
72 S(1,7)=C(1,1)*DUM1+C(1,2)*DUM2
C
C      MODIFY FOR ROLLER SUPPORT
C
300 IF(ITYPEINP(1,NE.1)) GO TO 301
   NP=NPJ
   KCOUNT=1
   MX=1
   GO TO 350
301 IF(ITYPEINPJ(NE.1)) GO TO 302
   NP=NPJ
   KCOUNT=2
   MX=3
   GO TO 350
302 IF(ITYPEINPK(NE.1)) GO TO 303
   NP=NPK
   KCOUNT=3
   MX=5
   GO TO 350
303 IF(NPL.EQ.0) GO TO 100
   IF(ITYPEINPL(NE.1)) GO TO 100
   NP=NPL
   KCOUNT=4
   MX=7
C
350 NX=MX+1
   CSN=DCOS(THETA(NP))
   SSN=OSIN(THETA(NP))
   DO 351 J=1,4
   S(J,MX)=S(J,MX)*CSN+S(J,NX)*SSN
351 S(J,NX)=0.0
C
   GO TO (301,302,303,100),KCOUNT
C
C      DISTRIBUTE ELEMENT STRESS TO NODE POINT STRESS
C
100 IF(ICOUNT.NE.1) GO TO 101
   MJ=1
   MJ=3
   MK=5
   ML=7

```

```

   NI=NPJ
   NJ=NPJ
   NK=NPK
   NL=NPL
   GO TO 103
C
101 IF(ICOUNT.NE.2) GO TO 102
   MI=3
   MJ=1
   MK=5
   ML=7
   NI=NPJ
   NJ=NPJ
   NK=NPK
   NL=NPL
   GO TO 103
C
102 IF(ICOUNT.NE.3) GO TO 107
   MI=5
   MJ=1
   MK=3
   ML=7
   NI=NPK
   NJ=NPJ
   NK=NPJ
   NL=NPL
   GO TO 103
C
107 MI=7
   MJ=1
   MK=3
   ML=5
   NI=NPL
   NJ=NPJ
   NK=NPJ
   NL=NPK
C
103 DO 114 I=1,4
   STNPU(1,NI)=STNPU(1,NI)+S(I,MI)
114 STNPU(1,NI)=STNPU(1,NI)+S(I,M(I+1))
   N=1
155 GO TO (150,151,152,153),N
150 NN=NJ
   MN=MJ
   GC TO 154
151 NN=NK
   MN=MK
   GO TO 154
152 IF(NPL.EQ.0) GO TO 153
   NN=NL
   MN=ML
C
154 DO 104 K=1,MXANDJP
   J=K
   IF(INPADJIN(K)-NPDUT).EQ.NV) GO TO 105
104 CONTINUE

```



```

1 STNPU(IST,1)=STNPU(IST,1)/DUM
  NPN=NADJNP(1)
  DO 8 J=1,NPN
    DO 9 IST=1,4
      STADU(IST,1,J)=STADU(IST,1,J)/DUM
    8 STADU(IST,1,J)=STADU(IST,1,J)/DUM
  9 RETURN
  END

SUBROUTINE LINK10
  IMPLICIT REAL*8(A-H,O-Z)
  COMMON MAXNP,MXADJP,MXLINE,MXLOAD,MXNPR,MXELB,MXCLB,NUMNP,NUMEL,
1 ISTRS,LINES,NUMPEL,LOADNP(1),PERIOD,OT,NPLOAO(1,50),
2 RAD(1,50),ZAO(1,50),SNORM(1,50),TITLE(10),MAXRO,
3 MBLOCK,NREAS,MAXMBK,TREAL,PDAMP,BETA,KRUN,
4 MXZONE,NZONES,IPRINT,NPTN,IELAST,MGT,EL,E2,E3,E4,E5
  DIMENSION NPTN(1000),IELAST(20),MGT(20),E1(20),E2(20),E3(20),E4(20),E5(20)
  DIMENSION NPLW(100),NPHIGH(100)
  BLOCK THE STIFFNESS AND STRESS TABLES
  SET MAXIMUM NUMBER OF CLUSTERS AND CALCULATE BLOCK SIZE
  MXCLS=100
  IF(NUMNP.LE.MAXMBK) MBLOCK=NUMNP
  IF(NUMNP.GT.MAXMBK) MBLOCK=MAXRO+1
  IF(KRUN.EQ.1) GO TO 1
  CALL SIZE(MXCLS,NUMCLS,NPLOW,NPHIGH,NUMNP,MBLOCK)
  WRITE TITLE RECORD ON STIFFNESS TAPE
  REWIND 10
  WRITE(10)TITLE
  1 CALL CLUSTER(MXCLS,NUMCLS,NPLOW,NPHIGH,NUMNP,MXADJP,NPTN,
1 MAXNP,PERIOD,KRUN)
  WRITE(6,1000)MBLOCK
1000 FORMAT(21HMATRIX BLOCK SIZE IS,15,6H NODES)
  RETURN
  END

SUBROUTINE SIZE(MXCLS,NUMCLS,NPLOW,NPHIGH,NUMNP,MBLOCK)
  IMPLICIT REAL*8(A-H,O-Z)
  DIMENSION NPLW(MXCLS),NPHIGH(MXCLS)
  COMPUTE THE SIZES OF THE BLOCKS
  NUMCLS=1
  NODES=0
  1 NPR=NUMNP-NODES
  NPLOW(NUMCLS)=NODES+1
  IF(NPR.GT.MBLOCK) GO TO 2
  NPHIGH(NUMCLS)=NUMNP
  RETURN
  2 NPHIGH(NUMCLS)=NODES+MBLOCK
  NODES=NUMCLS+1
  NUMCLS=NUMCLS+1
  IF(NUMCLS.LE.MXCLS) GO TO 1
  WRITE(6,1000)MBLOCK
1000 FORMAT(16H1ERROR IN LINK10 IN SUBROUTINE SIZE, TOO MANY CLUSTERS.
1 MBLOCK=,15,6H NODES)
  STOP
  END

SUBROUTINE CLSTER(MXCLS,NUMCLS,NPLOW,NPHIGH,NUMNP,
1 MXADJP,NPTN,MAXNP,PERIOD,KRUN)
  IMPLICIT REAL*8(A-H,O-Z)
  DIMENSION NPLW(MXCLS),NPHIGH(MXCLS),NPTN(MAXNP)
  DIMENSION NADJNP(100),ITYPE(100),XMASS(100),SNPUU(100),
1 SNPUU(100),SNPWW(100),NPAOJ(100,8),SAOUU(100,8),SADUM(100,8),
2 SADUM(100,8)
  3 SADUM(100,8)
  DIMENSION NADJEL(100),STNPU(4,100),STNPU(4,100,8),
1 STADU(4,100,8)
  EQUIVALENCE (STNPU(1,1),ITYPE(1)),(STNPU(1,26),THETA(1)),
1 (STNPU(1,51),XMASS(1)),(STNPU(1,76),SNPUU(1,1)),(STNPU(1,1)),
2 SNPUU(1,1),(STNPU(1,26),SNPWW(1,1)),STADU(1,1),SADUU(1,1),
3 (STADU(1,1,3),SADUM(1,1)),(STADU(1,1,5),SAOWW(1,1)),
4 (STADU(1,1,7),SAOWU(1,1))
  PERFORM THE BLOCKING OPERATION
  INITIALIZE AND SKIP TO SECOND PART FOR PRIMARY RERUN
  IF(KRUN.EQ.1) GO TO 300
  READ STIFFNESS TABLES FROM TAPE 3 AND WRITE BLOCKS ONTO TAPE 10
  REWIND 3
  ITCOUNT=1
101 NLOW=NPLW(ITCOUNT)
  NHGH=NPHIGH(ITCOUNT)
  DO 100 L=NLOW,NHGH
    I=L-NLOW+1
100 READ(3) NPN,NADJNP(1),ITYPE(1),THETA(1),XMASS(1),SNPUU(1),
1 SNPUU(1),SNPWW(1),NPAOJ(1,J),SADUU(1,J),SADUM(1,J),SADUM(1,J),
2 SADUM(1,J),J=1,MXADJP
  NUMNP=NHGH-NLOW+1
  WRITE(10)NLOW,NHGH,NUMNP,
1 (NADJNP(1),ITYPE(1),THETA(1),XMASS(1),SNPUU(1),SNPWW(1),
2 (NPAOJ(1,J),SADUU(1,J),SADUM(1,J),SAOWW(1,J),SADUM(1,J),J=1,MXADJP
3 ),I=1,NUMNP)

```



```

NUMCEL=0
NCLUST=0
C 100 IF((NUMDEL-NUMCEL).LT.MAXELB) NUMFLB=NUMDEL-NUMCEL
      IF((NUMDEL-NUMCEL).GE.MAXELB) NUMELR=MAXELR
C
C 200 KK=1,NUMELB
C
C  READ(14) NOOFEL(KK),LZONE,(PLAST(KK),NP(KK,1),NP(KK,2),NP(KK,3),
1  NP(KK,4),LTYPE(KK,1),LTYPE(KK,2),LTYPE(KK,3),LTYPE(KK,4),
2  THETA(KK,1),THETA(KK,2),THETA(KK,3),THETA(KK,4),
3  R(RJ,RK,RL,ZI,ZJ,ZK,ZL)
C
C  IF=1
201  NODE=NP(KK,1)
      NP(KK,1)=NPTP(NODE)
      I1=11+1
      IF(I1.LT.4) GO TO 201
      IF(I1.GT.4) GO TO 202
      IF(NPIKK,4).EQ.0) GO TO 202
      GO TO 201
C
C 202 IF=1ELAST(LZONE)
      A1=E1(LZONE)
      A2=E2(LZONE)
      A3=E3(LZONE)
      A4=E4(LZONE)
      A5=E5(LZONE)
      NUME=NOOFEL(KK)
      CALL ELAST(1E,1STRES,A1,A2,A3,A4,A5,CC,NUME)
      DO 203 I=1,4
      DO 203 J=1,4
203  C(KK,I,J)=CC(I,J)
C
      DO 204 I=1,4
      DO 204 J=1,8
      B(KK,I,J)=0.0
204  P(KK,J,I)=0.0
C
C  IF(NPIKK,4).NE.0) GO TO 208
      AJ=RJ-RI
      AK=RK-RI
      PJ=ZJ-ZI
      BK=ZK-ZI
      HH=AJ*PK-AK*RJ
      AA=AJ-AK
      BB=BK-BK
      R(KK,1,1)=BB/HH
      R(KK,1,3)=BK/HH
      B(KK,1,5)=-BJ/HH
      IF(1STRES.NE.0) GO TO 205
      RD=(AJ*AK)/3.
      ZO=(BJ*BK)/3.
      CAPRO=R(+RO)
      B(KK,2,1)=(HH+RR*RO-AA*ZO)/(HH*CAPRO)
      B(KK,2,3)=(BK*RO-AK*ZO)/(HH*CAPRO)
      B(KK,2,5)=(-BJ*RO+AJ*ZO)/(HH*CAPRO)
205  B(KK,3,2)=-AA/HH
      B(KK,3,4)=-AK/HH
      B(KK,3,6)=AJ/HH
      R(KK,4,1)=B(KK,3,2)
      B(KK,4,2)=B(KK,1,1)
      B(KK,4,3)=B(KK,3,4)
      B(KK,4,4)=B(KK,1,3)
      R(KK,4,5)=B(KK,3,6)
      R(KK,4,6)=B(KK,1,5)
C
      IF(1STRES.NE.0) CONST=HH/2.
      IF(1STRES.EQ.0) CONST=HH*CAPRO/Z.
C
212  DO 206 I=1,8
      DO 206 J=1,4
      DO 207 N=1,4
207  P(KK,I,J)=P(KK,I,J)+CONST*B(KK,N,1)*CC(N,J)
206  CONTINUE
      GO TO 231
C
C 208 AJ=RJ-RI
      BJ=ZJ-ZI
      AA=DSORT(AJ*AJ+RJ*RJ)
      AL=RL-RI
      RL=ZL-ZI
      RB=DSORT(AL*AL+BL*BL)
      HH=AA*RB
      SI=-BJ/AA
      CI=AJ/AA
C
      IF(1STRES.EQ.0) GO TO 209
      RO=AA/2.
      ZC=RB/2.
      GO TO 210
209  AINT1=HH
      AINT2=HH*RB/2.
      AINT3=HH*AA/2.
      AINT4=HH*RI+(AA*CI+RB*SI)/2.
      AINT13=AA*AINT4/2.+HH*AA**2*CI/12.
      AINT14=RB*AINT4/2.+HH*RB**2*SI/12.
      AINT16=(H/2.)*2
      RC=AINT13/AINT4
      ZO=AINT14/AINT4
      DUMMY=RO*SI+ZO*CI
      R(KK,1,1)={-AA*SI-RB*CI+DUMMY}/HH
      R(KK,1,3)={RB*CI-DUMMY}/HH
      R(KK,1,5)=DUMMY/HH
      B(KK,1,7)={AA*SI-DUMMY}/HH
      IF(1STRES.NE.0) GO TO 211
      R(KK,2,1)=(HH*AINT1-RB*AINT3+AINT16-AA*AINT2)/(HH*AINT4)
      R(KK,2,3)={BB*AINT3-AINT16}/(HH*AINT4)
      R(KK,2,5)=AINT16/(HH*AINT4)

```

```

      B(KK,2,7)=(AA*AIN2-A(NT16)/(HH*AIN4)
211  DUMMY=RO*C1-20*S1
      R(KK,3,2)=(RB*S1-AA*C1)+DUMMY)/HH
      R(KK,3,4)=(-BB*S1-DUMMY)/HH
      B(KK,3,6)=DUMMY /HH
      R(KK,3,8)=(AA*C1-DUMMY)/HH
      B(KK,4,1)=B(KK,3,2)
      B(KK,4,2)=B(KK,1,1)
      B(KK,4,3)=B(KK,3,4)
      B(KK,4,4)=B(KK,1,3)
      B(KK,4,5)=B(KK,3,6)
      B(KK,4,6)=B(KK,1,5)
      B(KK,4,7)=B(KK,3,8)
      B(KK,4,8)=B(KK,1,7)
      IF(ISTRES.NE.0) CONST=HH
      IF(ISTRES.EQ.0) CONST=AIN4
      GO TO 212
C
C
231  DO 232 I=1,4
      EPSTII(KK,I)=0.0
232  S(GII(KK,I))=0.0
C
C
      IF(IPLAST(KK).NE.1) GO TO 237
      IDUM(KK,21)=NOVLO(IZONE)
      KYILO=NOVLO(IZONE)
      DO 233 I=1,KYILO
        DUM(KK,I)=SSTAR(IZONE,I)
233  DUM(KK,I+10)=HSTAR(IZONE,I)
      DO 234 I=1,8
        DUM(KK,I+21)=0.0
234  DUM(KK,27)=SSTAR(IZONE,I)
      GO TO 200
C
C
237  IF(IPLAST(KK).NE.2) GO TO 235
      DUM(KK,1)=ALPHA(IZONE)
      DUM(KK,2)=CAPPA(IZONE)
      DUM(KK,3)=COSTH(IZONE)
      IDUM(KK,4)=0
      IF(CAPPA(IZONE).EQ.0.0) IDUM(KK,4)=1
      DUM(KK,5)=0.0
      DO 238 I=1,4
        DUM(KK,I+5)=0.0
238  GO TO 200
C
C
235  WRITE(6,236) NOOFEL(K),IZONE,IPLAST(KK)
236  FORMAT(1H,28HERROR IN ELEMENT DATA, LNKIF//
      10X,12HELEMENT NO.=,15/10X,12HIZONE NO.
      20X,12HPLAST =,15)
      CALL EXIT
C
C
200  CONTINUE

```

```

C
C
      NUMCEL=NUMCEL+NUMELB
      WRITE(12) NUMCEL,NUMELB,(NOOFEL(K),IPLAST(K),(NP(K,J),(TYPE(K,J),
      1THETA(K,J),J=1,4),(C(K,J,I),I=1,4),(B(K,J,I),(=1,8),J=1,4),
      2(IP(K,J,I),J=1,8),EPSTII(K,I),EPSPII(K,I),S(GII(K,I),I=1,4),
      3(OUT(K,I),(=1,4),K=1,NUMELB)
      NCLUST=NCLUST+1
C
      IF(NUMCEL.LT.NUMPEL) GO TO 100
C
      ENO FILE 12
      REWIND 12
      REWIND 14
C
      311 CONTINUE
      RETURN
      END

```

```

SUBROUTINE ELOST(IELAST,ISTRES,F1,F2,E3,E4,F5,C,NUME)
IMPLICIT REAL*8(A-H,O-7)
DIMENSION C(4,4)

```

C CALCULATE THE PLASTIC STRESS-STRAIN MATRIX

```

C
C
      DO 1 I=1,4
      DO 1 J=1,4
      C(I,J)=0.0
C
      IF(IELAST.NE.1) GO TO 20
C
      IF(ISTRES.EQ.2) GO TO 4
C
      EBAR=E1/(1.+E2)*(1.-2.*E2)
      C(1,1)=EBAR*(1.-E2)
      C(1,2)=EBAR*E2
      C(1,3)=C(1,2)
      C(2,1)=C(1,2)
      C(2,2)=C(1,1)
      C(2,3)=C(1,2)
      C(3,1)=C(1,2)
      C(3,2)=C(1,2)
      C(3,3)=C(1,1)
      C(4,4)=EBAR*(1.-2.*E2)/2.
      RETURN

```

```

C
      4  EBAR=E1/(1.-E2*E2)
      C(1,1)=EBAR
      C(3,1)=EBAR*E2
      C(1,3)=C(3,1)
      C(3,3)=C(1,1)
      C(4,4)=EBAR*(1.-E2)/2.
      RETURN

```

C 20 IF(IELAST.NE.2) GO TO 30

```

21 WRITE(6,3) IELAST,NUME,ISTRES
3  FORMAT(11H/31H ERROR IN ELASTIC CONSTANT DATA/
113H IELAST =,15/13H ELEMENT NO.=,15/
213H ISTRES =,15)
CALL EXIT
C
30 IF(IELAST.NE.3) GO TO 21
C
IF(ISTRES.EQ.2) GO TO 21
C
DO 31 I=1,3
DO 31 J=1,3
31 C(I,J)=E1
C
RETURN
END
C
SUBROUTINE LINK2
IMPLICIT REAL*8(A-H,O-Z)
COMMON MAXNP,MXADJP,MXLINE,MXLOAD,MXNPB,MXELR,NUMNP,NUMEL,
1  ISTRES,LINES,NUMPEL,LOADNP(1),PERIOD,OT,NPLOAD(1,50),
2  RAD(1,50),ZAD(1,50),SNORM(1,50),TITLE(10),MAXBD,
3  MBLOCK,NREADS,MAXMRK,TRFAL,PDAMP,RETA,KRUN,
4  IPELTP,KOUNT,IFILLA,T,TMAX,TRERUN
DIMENSION COM(198)
EQUIVALENCE (MAXNP,COM(1))
C
THIS LINK INTEGRATES THE EQUATIONS OF MOTION
C
READ AND PRINT INTEGRATION DATA
C
READ(5,1000)TMAX,TRERUN,ET,KDT,KINT,BETA,PDAMP
1000 FORMAT(3E10.0,2I5,2E10.0)
C
TMAX =MAX. RUN TIME FOR PROBLEM (SEC.)
TRERUN=BEGINNING TIME FOR RESTART (SEC.)
ET =TIME INCREMENT (SEC.)
KDT =0 USES DT AS READ IN
=1 USES KINT TO CHOOSE DT
KINT =DIVISOR FOR CHOOSING DT
BETA =INTEGRATION SCHEME PARAMETER
PDAMP =PERCENT CRITICAL DAMPING FOR ARTIFICIAL VISCOSITY
C
IF(BETA.LE.0.0-OR.BETA.GT.0.25) BETA=1.0/6.0
REWRITE
IF(KRUN.LT.2) GO TO 10
REACH(8)ITITLE
READ(8)COM
GO TO 20
10 WRITE(8)ITITLE
DT=ET
IF(KDT.NE.0) DT=PERIOD/FLOAT(KINT)
NREADS=1
IF(NUMPEL.NE.0) NREADS=2+(NUMPEL-1)/MAXFLB
TREAL=0.0
C
WRITE(8)COM
20 WRITE(6,2000)TMAX,TRERUN,OT,BETA,PDAMP
2000 FORMAT(1H1,30HMAX. TIME DURATION
1 1X,30HINITIAL TIME FOR INTEGRATION
3 1X,30HTIME INCREMENT
7 1X,30HRETA
8 1X,30HPOAMP
C
CHECK LIMITS
C
IF(KRUN.EQ.0-AND.TMAX.EQ.0.0) STOP
40 IF(MBLOCK.LE.MAXMRK) GO TO 50
WRITE(6,3000)MBLOCK,MAXMRK
3000 FORMAT(49H1BANDWIDTH TOO LARGE FOR INTEGRATION CALCULATION./
114H BANDWIDTH IS ,15,31H, MAXIMUM BANDWIDTH ALLOWED IS ,15)
STOP
C
CALCULATE MATRICES FOR RECURSION
C
50 IF(KRUN.GT.1) GO TO 60
CALL LINK2A
C
INTEGRATE PROBLEM
C
60 IF(TMAX.NE.0.0) GO TO 70
END FILE B
REWIND B
STOP
70 CONTINUE
CALL LINK2B
C
RETURN
END
C
SUBROUTINE MM7IA,L,M,N,B,IA,IB,C)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(L,M),R(M,N),C(L,M)
C
MATRIX MULTIPLY WITH TRANSPOSE OPTION
C
DO 10 IL=1,L
DO 10 IN=1,N
DO 10 IM=1,M
11=IL
IF(IA.LT.0) 11=IM
12=IL+IM-11
13=IM
IF(IB.LT.0) 13=IN
14=IM+IN-13
C(IL,IN)=C(IL,IN)+A(11,(2)*R(13,14)
RETURN
END
10
C
SUBROUTINE ERASE(N,X)

```

```

REAL*8 X(N)
ERASE N WORDS IN X
DO 10 I=1,N
  10 X(I)=0.0
RETURN
ENC

SUBROUTINE LINK2A
  IMPLICIT REAL*8(A-N,O-Z)
  COMMON MAXNP,MXADJP,MXLINE,MXLOAD,MXNPR,MXFLR,MXNMP,NUMPL,
  1 ISTRS,LINES,NUMPEL,LOADNP(1),PERION,DT,NPLOAD(1,50),
  2 MAO(1,50),ZAO(1,50),SNORM(1,50),TITLE(10),MAXBO,
  3 MBLOCK,NREAS,MAXWBK,TRFAL,POAMP,BETA,KRUN,
  4 IPELTP,KOUNT,IFILL,A,T,THAX,TPERUN
  DIMENSION NADJNP(30),ITYPE(30),THETA(30),XMASS(30),SNP(30,3),
  1 NPAOJ(30,8),SAD(30,8,4),F(4,30,3D),F(4,30,30),TMP(4),OMP(2,30),
  2 DIMENSION STRESS(4,6,50),NPARJL(50,6),NADJPL(50),QI(4,50,50),
  3 IRI(4,30,100),QI(4,50,50),SPACE(2000),
  4 DIMENSION KEYS(3,100)
  EQUIVALENCE (F(1,1,1),SPACE(1)),(F(1,1,1),SPACE(4001)),
  1 (QI(1,1,1),SPACE(1)),(QI(1,1,1),SPACE(10001)),
  2 (IRI(1,1,1),SPACE(4001))

  CALCULATE MATRICES FOR RECURSION AND SAVE ON TAPE

  INITIALIZE
  REWIND 10
  READ(10)
  REWIND 20
  WRITE(20) TITLE
  MRECS=0

  READ ANOTHER BLOCK FROM STIFFNESS TAPE
  100 READ(10)N1,N2,N5,(NADJNP(I),ITYPE(I),THETA(I),XMASS(I),SNP(I,J),
  1 J=1,3),(NPAOJ(I,J),(SAD(I,J,K),K=1,4),J=1,MXADJP),I=1,N5)

  ZERO OUT FI

  CALL ERASF(4*MAXWBK*MAXWBK,FI)

  COPY C(N) INTO FI

  DO 950 I=1,N5
    950 I=N1-I

  ADD DIAGONAL SFT OF EQUATIONS
  OAMU=2.0*DSORT(XMASS(1)*SNP(1,1))*POAMP/100.0
  OMP(1,1)=OAMU
  UAMU=2.0*DSORT(XMASS(1)*SNP(1,3))*PDAMP/100.0

```

```

  DMP(2,1)=DAMU
  FI(1,1,1)=(O.5*OAMU*OT+XMASS(1))/(BETA*OT*OT)+SNP(1,1)
  FI(2,1,1)=SNP(1,2)
  FI(3,1,1)=SNP(1,2)
  FI(4,1,1)=(C.5*OAMU*OT+XMASS(1))/(BETA*OT*OT)+SNP(1,3)

  CHECK ADJACENT STIFFNESS TABLES FOR NODES IN THIS BLOCK

  DO 945 K=1,MXADJP
    JNP=NPAOJ(I,K)
    IF(JNP.LT.N1.OR.JNP.GT.N2) GO TO 945
    J=JNP-N1+1

  COPY ADJACENT STIFFNESS INTO FI AND ZERO IT OUT IN TABLES

  FI(1,1,J)=SAD(I,K,1)
  FI(2,1,J)=SAD(I,K,3)
  FI(3,1,J)=SAD(I,K,2)
  FI(4,1,J)=SAD(I,K,4)
  DO 920 M=1,4
    920 SAD(I,K,M)=0.0
    NACJNP(I)=NADJNP(I)-1
    NPAOJ(I,K)=0

  ZERO OUT ITS TRANSPOSE ACROSS THE MAIN DIAGONAL

  DO 925 XI=1,MXADJP
    KK=XI
    IF(NPAOJ(J,K).EQ.INP) GO TO 930
    925 CONTINUE
    WRITE(6,1000)INP,JNP
  1000 FORMAT(36HITABLE SEARCH ERROR IN LINK2A. INP=,I5,6H, JNP=,I5)
  STOP
  930 OC'S35 M=1,4
  935 SAD(I,K,K,M)=0.0
  NACJNP(J)=NADJNP(J)-1
  NPAOJ(J,KK)=0
  945 CONTINUE
  950 CONTINUE

  SKIP FOR FIRST BLOCK
  IF(N1.EQ.1) GO TO 150

  T
  SUBTRACT OFF C(N) *F(N-1)*C(N)

  LOOP OVER EACH NOOF IN FI

  DO 120 I=1,N5
    DO 120 J=I,N5

  LOOP OVER EACH NODE IN C
  DO 120 KI=1,MXADJP
    L=NPAOJ(I,KI)

```



```

190 CONTINUE
C
C   WRITE (M) ON TAPE
C
205 WRITE(20)M1,N2,N5,(IF(1,J,K),(-1,4),K=J,N5(J=1,N5(
C
C   WRITE REPT OF TABLES ON TAPE
C
WRITE(20)(ITYPE(I),THEYAL(I,XMASSII),(IMPJ,I),J=1,2),I=1,N5(
IFIM1,FO,1) GO TO 225
DO 215 I=1,N5
N7=N5-1+I
IFIMADJMP(7),MF,6) GO TO 227
215 CONTINUE
227 WRITE(20)N7,IMPADJMP(1,IMPADJII,J),ISADII,J,K(=1,4),J=1,MAXADJ(
11=1,N7)
225 N1LAST=N1
C
C   SKIP FOR LAST BLOCK
C
IFIM2,FO,MUNMP) GO TO 507
C
WRITE (M)C(M+1) ON TAPE
C
CALL DISK(12,MRECS+1,FI,4,MAXIMPRKMAXM(K)
KEYS11,MRECS+1,M1
KEYS12,MRECS+1,M5
KEYS13,MRECS+1,M6
MRECS=MRECS+1
GO TO 100
C
C   READ BACK RECORDS FROM TAPE IN REVERSE ORDER
C
507 IFIMRECS,FO,FI) GO TO 501
DO 510 M7=1,MRECS
CALL DISK(1,MRECS+1-M7,FI,4,MAXIMPRKMAXM(K)
M1=KEYS11,MRECS+1-M7)
M5=KEYS12,MRECS+1-M7)
M6=KEYS13,MRECS+1-M7)
510 WRITE(20)M1,N2,N5,IMPADJII,J,K(=1,4),J=1,N5(K=1,N6)
C
C   CALCULATE STRESS CONSTRAINT TABLES
C
501 REMING 20
PFAD(1)
M1AD=LOADMP(1)
600 REACT(1)M1,N2,N5,IMPADJMP(1),IMPADJII,J,J=1,MAXADJ(
1(OIK,1,1),OIK,1,2),K=1,4),I(OIK,1,J),RIK,1,J,K(=1,4),
2J=1,MAXADJ(1,1,N5)
C
C   REMING 10
PFAD(20)
IMALF=1
JHALF=51
MRECS=0
650 PFAD(20)M1,N2,N5,1(IF(1,J,K),1=1,4),K=J,N5),J=1,N5)
PFAD(20)ITYPE(1),THEYAL(1,XMASSII),IMPADJII,J,K(=1,4),J=1,N5)
IFIM1,FO,1) GO TO 655
PFAD(20)M7,IMPADJMP(1,IMPADJII,J),ISADII,J,K(=1,4),J=1,MAXADJ(
11=1,N7)
655 CALL FRASE(4,MAXIMPRKMAXM(K),1,1,JHALF)
C
C   PUT IN GIN
C
DO 671 I=1,NLOAD
10JMM1=1+JHALF-1
NUM=MAXADJ(11)
JMODE=IMPADJ(1,J)
IF(JMODE,LT,M1,OR,JMODE,GT,M2) GO TO 677
P11,JMODE=M1+1,10JMM1 (=STRESS11,J,1)
P12,JMODE=M1+1,10JMM1 (=STRESS12,J,1)

```


[illegible]

END

SUBROUTINE LINK3

IMPLICIT REAL*8(A-H,O-Z)

COMMON MAXNP,MXADJ,MXLINE,MXLOAD,MXNPR,MXFLR,MXFLR,MUMNP,MUMFL,
1 ISTRS,LINES,NUMPEL,LOADNP(1),PERIOD,OT,NPLOAD(1,50),
2 RAD(1,50),ZAD(1,50),SNORM(1,50),TITLE(10),MAXRD,
3 MRLDCK,NREADS,MAXMRK,TREAL,PDAMP,BETA,KRUN,
4 NUMOUT,TMAX,TSTRSS,IOUT,JOUT
DIMENSION UDDN(100),WDDN(100),UDN(100),WDN(100),UN(100),
1 WNI(100),PRESSU(1,100),PRESSW(1,100),FILL(200)
DIMENSION STNPU(4,100),STNPM(4,100),STADU(4,100,8),
1 MADJEL(100),MADJNP(100),MPADJ(100,8),NPOUT(100),NPTP(100)
DIMENSION BLOCKA(320),BLOCKB(320)
EQUIVALENCE (UDN(1),BLOCKA(1)),(WDDN(1),BLOCKA(101)),(UDN(1),
1 BLOCKA(201)),(PRESSU(1,1),BLOCKA(301)),(PRESSW(1,1),BLOCKA(310))
2),(WNI(1),BLOCKB(1)),(UN(1),BLOCKB(101)),(WNI(1),BLOCKB(201)),
3(FILL(1),BLOCKB(301))
DIMENSION COM(198)
EQUIVALENCE(MAXNP,COM(1)).

THIS LINK PROCESSES THE DTH TAPE AND CALCULATES STRESSFS,
PRINTS AND WRITES BOTH ON THE DTH TAPE

REWIND 8

POSITION DTH TAPE PAST TITLE AND COMMON BLOCKS

READ(8)TITLE
READ(8)COM

CALL LINK3A(BLOCKA,BLOCKB,STNPU,STNPM,STADU,STADW,MADJEL,MADJNP,
1 MPADJ,NPOUT,NPTP)
CALL LINK3B(UDN,WDDN,UDN,WNI,PRESSU,PRESSW,STNPU,STNPM,
1 STADU,STADW,MADJEL,MADJNP,NPADJ,NPOUT,NPTP)

RETURN
END

SUBROUTINE LINK3A(TTADU,TTADW,STNPU,STNPM,STADU,STADW,MADJEL,
1 MADJNP,NPADJ,NPOUT,NPTP)
IMPLICIT REAL*8(A-H,O-Z)
COMMON MAXNP,MXADJ,MXLINE,MXLOAD,MXNPR,MXFLR,MXFLR,MUMNP,MUMFL,
1 ISTRS,LINES,NUMPEL,LOADNP(1),PERIOD,OT,NPLOAD(1,50),
2 RAD(1,50),ZAD(1,50),SNORM(1,50),TITLE(10),MAXRD,
3 MRLDCK,NREADS,MAXMRK,TREAL,PDAMP,BETA,KRUN,
4 NUMOUT,TMAX,TSTRSS,IOUT,JOUT
DIMENSION STNPU(4,100),STNPM(4,100),STADU(4,100,8),STADW(4,100,8),
1 MADJEL(100),MADJNP(100),MPADJ(100,8),NPOUT(100),NPTP(100)
DIMENSION NPTN(100),ANAME(6),MADJNP(100),MADJEL(100),
1 MPADJ(100,8),TTNPU(4,100),TTNPM(4,100),TTADU(4,100,8),
2 TTADW(4,100,8)

READ CARD INPUT AND STRESS TABLES OFF BLOCKED STIFF. & STRESS TAPE

POSITION STRESS TAPE PAST TITLE RECORD

REWIND 10
READ(10)

POSITION STRESS TAPE PAST STIFF. TABLES AND NPTN RECORD

1 READ(10) N1,N2
(F(1,2),LT,NUMNP) GO TO 1
READ(10) (NPTN(1),I=1,NUMNP)

READ AND PRINT CARD INPUT

READ(5,2)TSTRSS,TMAX,NUMOUT,IOUT,JOUT
IF(TMAX.EQ.D.D) TMAX=1.E+20
IF(IOUT.EQ.D) IOUT=1
IF(JOUT.EQ.D) JOUT=1

READ(5,3) (NPOUT(1),I=1,NUMOUT)
2 FORMAT(2E10,D,3I5)
3 FORMAT(14I5)

TSTRSS=BEGINNING TIME FOR RECOMPUTING OUTPUT (SEC.)

TMAX =MAXIMUM OUTPUT TIME (SEC.)

NUMOUT=NO. OF OUTPUT NDDDES (LE,100)

IOUT =NO. OF INTEGRATION POINTS PER PRINTED OUTPUT POINT

JOUT =NO. OF INTEGRATION POINTS PER TAPE OUTPUT POINT

NPOUT =OUTPUT NDDDE NUMBERS

WRITE(6,4)TSTRSS,TMAX,NUMOUT,IOUT,JOUT

4 FORMAT(19H1OUTPUT START TIME=,1PE15.5/
1 18H OUTPUT STOP TIME=,1PE15.5/
2 21H NO. OF OUTPUT NDDDES=,15/
3 25H PRINTED OUTPUT INTERVAL=,15/
4 22H TAPE OUTPUT INTERVAL=,15///
5 20H OUTPUT NDDDE NUMBERS)
WRITE(6,5) (NPOUT(1),I=1,NUMOUT)
5 FORMAT(10X,5I5)

CONVERT TO NEW NUMBERING SYSTEM AND CHECK FOR ERRORS

DO 6 1=1,NUMOUT

NPOLD=NPOUT(1)

DO 7 J=1,NUMNP

JJ=J

IF(NPTN(J).EQ.NPOLD) GO TO 6

7 CONTINUE

WRITE(6,9) NPOLD,(NPTN(J),J=1,NUMNP)

9 FORMAT(1H1,14HERROR IN SETUP//10X,5//10X,15I5))

CALL EXIT

8 NPTP(1)=JJ

6 CONTINUE

SKIP FOR SUMMARY RFRUNS

IF((KRUN.EQ.3).OR.(KRUN.EQ.4)) GO TO 100


```

403 IF(1C.GT.IOUT) 1C=1
    IF(JC.GT.JOUT) JC=1
    GO TO 400
999 CONTINUE

```

C

```

END FILE 3
REWIND 3
RETURN
END

```

```

SUBROUTINE ARCTAN(X,Y,THETA)
IMPLICIT REAL*8(A-H,O-Z)
DATA PI/3.141592653589793/

```

C

```

C      GET ARC TANGENT IN CORRECT QUADRANT BETWEEN
C      +PI AND -PI

```

C

```

C      IF(X)50,10,50
C      IF(Y)40,30,20
C      THETA=PI/2.0

```

10

```

RETURN

```

20

```

THETA=0.0D0

```

30

```

THETA=-PI/2.0

```

40

```

RETURN

```

50

```

THETA=ATAN(DABS(Y/X))

```

60

```

IF(X)90,60,60

```

70

```

IF(Y)70,80,80

```

80

```

THETA=-THETA

```

90

```

IF(Y)110,100,100

```

100

```

THETA=PI-THETA

```

110

```

THETA=-PI+THETA

```

```

RETURN

```

```

END

```

```

00111150
00111200

```

```

00111250
00111300
00111350
00111400

```

```

00111450
00111500

```

```

00111600
00111650
00111700

```

```

00111800
00111850
00111900
00111950

```

```

00112000
00112050
00112100

```

```

00112200

```

```

00112300
00112350

```

Reproduced from
best available copy.

APPENDIX A1. (MINBND) SOURCE LISTING.

```

SUBROUTINE MINBND(IROW,ICOL,NUMN,IS,JS,MBW)
  DIMENSION IROW(1),ICOL(1),NAP(1),IS(1),JS(1)
  C
  C MATRIX BANDWIDTH MINIMIZATION - - KUSEN 11968)
  C
  NCP=0
  REMIND 8
  DO 20 I=1,NUMN
    READ 101 IS(I),NAP(I),K=1,NAI)
    DO 20 J=1,NAI
      IF(I-NAP(J))10,20,20
    10 NCP=NCP+1
  20 CONTINUE
  REMIND 8
  CALL NUMBND(IROW,ICOL,IS,NCP,NUMN,MBW)
  DO 30 I=1,NUMN
    K=IS(I)
    JS(I)=J
  30 JS(I)=J
  RETURN
  END
SUBROUTINE NUMBND(IROW,ICOL,IS,NCP,NSIZE,MAXBND)
  DIMENSION IROW(1),ICOL(1),IS(1),INTPH2(2002)
  MBDPRV=NSIZE
  MAXPH2=NSIZE
  1 MAXBND=0
  DO 2 I=1,NCP
    JULFF=IROW(I)-ICOL(I)
    IF(JDIFF-MAXBND)2,2,1000
  1000 MAXBND=JDIFF
    ICMXBD=IROW(I)
    ICMXBD=ICOL(I)
  2 CONTINUE
  PRINT 6000, MAXBND
  6000 FORMAT(16H MAXBANDWIDTH = ,I5)
  IF(MAXBND-1)1001,1004,1001
  1001 ISTART=XMAXOF(1,(2=ICMXBD-ICMXBD>1))
    ISTOP=ICMXBD-1
    JJ=ICMXBD
  3 CONTINUE
    IF(I-ICMXBD)1002,3,1002
  1002 I1=1
    CALL SEARCH(IROW,ICOL,ICMXBD,I,NCP,NFUUND,MAXBND)
    IF(NFUUND-1)3,6,3
  3 CONTINUE
    ISTART=ICMXBD+1
    ISTOP=XMINOF(NSIZE,(2=ICMXBD-ICMXBD-1))
    JJ=ICMXBD
    DO 4 I=1,ISTART,ISTOP
      IF(I-ICMXBD)1003,4,1003
  1003 I1=1
    CALL SEARCH(IROW,ICOL,ICMXBD,I,NCP,NFUUND,MAXBND)
    IF(NFUUND-1)4,6,4
  4 CONTINUE
  1022 IF(MAXBND-MBDPRV)1010,7,1010
  1010 IEND=2+MAXPH2
    DO 8 I=1,IEND
      INTPH2(I)=0
      INTPH2=0
  8 INTPH2=0
  INTPH2=0
  END

```

```

  MBDPRV=MAXBND
  GO TO 9
  7 IF INTPH2-MAXPH2)9,1004,9
  1004 RETURN
  9 CONTINUE
  N2INT=2+INTPH2
  INTPH2=INTPH2+1
  ISTART=XMAXOF(1,(2=ICMXBD-ICMXBD))
  ISTOP=ICMXBD-1
  JJ=ICMXBD
  DO 31 I=1,ISTART,ISTOP
    IF(I-ICMXBD)1005,31,1005
  1005 I1=1
    CALL P2SRCH(IROW,ICOL,ICMXBD,I,NCP,NFUUND,MAXBND,INTPH2,N2INT)
    IF(NFUUND-1)31,6,31
  31 CONTINUE
    ISTART=ICMXBD+1
    ISTOP=XMINOF(NSIZE,(2=ICMXBD-ICMXBD))
    JJ=ICMXBD
    DO 44 I=1,ISTART,ISTOP
      IF(I-ICMXBD)1006,44,1006
  1006 I1=1
    CALL P2SRCH(IROW,ICOL,ICMXBD,I,NCP,NFUUND,MAXBND,INTPH2,N2INT)
    IF(NFUUND-1)44,6,44
  44 CONTINUE
    RETURN
  6 CALL INTRCH(IROW,ICOL,JJ,I1,NCP)
    M1=JJ
    M2=11
    DO 37 I=1,NSIZE
      IF(IS(I)-M1)1008,1007,1008
  1007 J1=1
  1008 IF(IS(I)-M2)37,1004,37
  1009 J2=1
  37 CONTINUE
    IS(J1)=M2
    IS(J2)=M1
    GO TO 1
  END
SUBROUTINE INTRCH(IROW,ICOL,IIS,IWAS,NCP)
  DIMENSION IROW(1),ICOL(1)
  DO 1 I=1,NCP
    IF(IROW(I)-IIS)2,5000,2
  5000 CALL SWITCH(IROW(I),ICOL(I),IWAS,JH,JL)
    IROW(I)=JH
    ICOL(I)=JL
    GO TO 1
  2 IF(IROW(I)-IWAS)3,5001,3
  5001 CALL SWITCH(IROW(I),ICOL(I),IIS,JH,JL)
    IROW(I)=JH
    ICOL(I)=JL
    GO TO 1
  3 IF(ICOL(I)-IIS)4,5002,4
  5002 CALL SWITCH(ICOL(I),IROW(I),IWAS,JH,JL)
    IROW(I)=JH
    ICOL(I)=JL
    GO TO 1
  4 IF(ICOL(I)-IWAS)1,5003,1
  5003 CALL SWITCH(ICOL(I),IROW(I),IIS,JH,JL)
    IROW(I)=JH

```

```
ICOL(1)=JL
1 CONTINUE
RETURN
```

```
END
```

```
SUBROUTINE P2SRCHT(IROW,ICOL,IC,JC,NCP,NF,MAXBND,INTPH2,N2INT)
DIMENSION IROW(1),ICOL(1),INTPH2(1)
```

```
NF=0
```

```
DO 1 I=1,NCP
```

```
IF(IROW(I)-IC12,4000,2
```

```
4000 ICAN2=XABSFC(JC-ICOL(I))
```

```
IF(ICAN2-MAXBND)1,1,4001
```

```
4001 RETURN
```

```
2 IF(ICOL(I)-JC13,4002,3
```

```
4002 NOCAN=XABSFC(IC-IROW(I))
```

```
IF(NOCAN-MAXBND)1,1,4001
```

```
3 IF(IROW(I)-JC14,4003,4
```

```
4003 IDO2=XABSFC(IC-ICOL(I))
```

```
IF(IDO2-MAXBND)1,1,4001
```

```
4 IF(ICOL(I)-IC11,4004,1
```

```
4004 NODU2=XABSFC(JC-IROW(I))
```

```
IF(NODU2-MAXBND)1,1,4001
```

```
1 CONTINUE
```

```
IF(N2INT)4005,5,4005
```

```
4005 MTI=0
```

```
MTJ=0
```

```
DO 6 I=1,N2INT
```

```
IF(INTPH2(I)-IC)61,4006,61
```

```
4006 MTI=1
```

```
61 IF(INTPH2(I)-JC)6,4007,6
```

```
4007 MTJ=1
```

```
6 CONTINUE
```

```
4008 IF(MTI<MTJ-2)5,4001,5
```

```
5 CONTINUE
```

```
NF=1
```

```
INTPH2(N2INT+1)=IC
```

```
INTPH2(N2INT+2)=JC
```

```
RETURN
```

```
END
```

```
SUBROUTINE P2SRCHT(IROW,ICOL,IC,JC,NCP,NF,MAXBND)
DIMENSION IROW(1),ICOL(1)
```

```
NF=0
```

```
DO 1 I=1,NCP
```

```
IF(IROW(I)-IC12,3000,2
```

```
3000 ICAN=XABSFC(JC-ICOL(I))
```

```
IF(ICAN-MAXBND)1,3001,3001
```

```
3001 RETURN
```

```
2 IF(ICOL(I)-JC13,3002,3
```

```
3002 NOCAN=XABSFC(IC-IROW(I))
```

```
IF(NOCAN-MAXBND)1,3001,3001
```

```
3 IF(IROW(I)-JC14,3003,4
```

```
3003 IDO=XABSFC(IC-ICOL(I))
```

```
IF(IDO-MAXBND)1,3001,3001
```

```
4 IF(ICOL(I)-IC11,3004,1
```

```
3004 NODU=XABSFC(JC-IROW(I))
```

```
IF(NODU-MAXBND)1,3001,3001
```

```
1 CONTINUE
```

```
NF=1
```

```
RETURN
```

```
END
```

```
SUBROUTINE P2SRCHT(IROW,ICOL,IC,JC,NCP,NF,MAXBND,JLUM)

```

```
JSM=JST
IF(ICM-JSW)2001,2000,2001
2000 JSM=ISW
2001 JHIGH=MAXOF(ICM,JSM)
JLOW=MINOF(ICM,JSM)
RETURN
END
```

5.0a

APPENDIX B. ØS/360 SLAMCODE LINK EDITOR CONTROL CARDS.

INSERT MAIN
 OVERLAY ALPHA
 INSERT LINK1
 OVERLAY BETA
 INSERT LINK1A,LOADIT,LINK1A,ORDER,ADJNR,VALDJR,PAFM,MINI,GSORTS,SORT1
 OVERLAY BETA
 INSERT LINK1B,ELAST,STIFF,INS,DISTN,ADJUST,MASS,PRNK
 OVERLAY BETA
 INSERT LINK1C,STRESS,MODS,AVG
 OVERLAY BETA
 INSERT LINK1D,SIZE,CLSTER
 OVERLAY BETA
 INSERT LINK1F,ELUST
 OVERLAY ALPHA
 INSERT LINK2,MMT,ERASE
 OVERLAY GAMMA
 INSERT LINK2A,DISKER,ESC
 OVERLAY GAMMA
 INSERT LINK2B,MSTRES
 OVERLAY ALPHA
 INSERT LINK3
 OVERLAY KHU
 INSERT LINK3A
 OVERLAY KHU
 INSERT LINK3H,AKCTAN
 ENTRY MAIN

A/3

APPENDIX C. OS/360 FORTRAN H SLAMCODE JOB CONTROL LANGUAGE CARDS.

a.) Requires two 9-track tape drives.

```
//LUN,
//EXEC PML0,PARM0,DEJECT=1,LIST,IMP,INLY,
//OBJECT, INPUT (D) UNIT=2400, VOL=SEMIPT1200, DSN=SLAM00
//DATA, FT01F001 (D) UNIT=SYSUA, SPACE=(1364,125),
//
//      ICB=(RECFM=VS, LRECL=37, BLKSIZE=364)
//DATA, FT03F001 (D) UNIT=(SYSUA,SEMIPT01F001), SPACE=(LVL,10),
//
//      DISP=(NEW,PASS),DSN=SL01T01,
//      ICB=(RECFM=VS, LRECL=3404, BLKSIZE=3408)
//DATA, FT04F001 (D) UNIT=(SYSUA,SEMIPT01F001,PT03F001),
//
//      SPACE=(364,100),
//      ICB=(RECFM=VS, LRECL=36, BLKSIZE=364)
//DATA, FT08F001 (D) UNIT=(SYSUA,SEMIPT01F001,PT03F001),
//
//      SPACE=(1484,100), ICB=(RECFM=VS, LRECL=448, BLKSIZE=448)
//DATA, FT10F001 (D) UNIT=2400, VOL=SEMIPT1200, DSN=ST1F01, LABEL=1,
//
//      ICB=(RECFM=VS, LRECL=2052, BLKSIZE=2056)
//DATA, FT12F001 (D) UNIT=(SYSUA,SEMIPT03F001,PT04F001),
//
//      SPACE=(1804,125), ICB=(RECFM=VS, LRECL=180, BLKSIZE=1804)
//DATA, FT14F001 (D) DDONLY
//DATA, FT20F001 (D) UNIT=SYSUA, SPACE=(LVL,20), DSN=SL01T01,
//
//      ICB=(RECFM=VS, LRECL=2052, BLKSIZE=2056)
//DATA, FT22F001 (D) UNIT=SYSUA, SPACE=(LVL,20)
//DATA, INPUT (D) "
```

A-1.4

Reproduced from
best available copy.

APPENDIX C. ØS/360 FORTRAN H SLAMCODE JOB CONTROL LANGUAGE CARDS.

b.) Requires four 9-track tape drives.

```

//*LIST
// EXEC PGM=SLAMCODE,OBJJECT='LIST,COMP,ONLY'
//DDJECT. INPUT DD UNIT=2400,VOL=SER=6P1Z,DD,DS=SLAMCODE
//DATA. FT01F001 DD UNIT=SYS0A,SPACE=(360,120),
// DCB=(RECFM=VS,LRECL=360, LKSIZ=364)
//DATA. FT03F001 DD UNIT=2400,VOL=SER=6P1Z,DD,DS=01F01APE,
// DCB=(RECFM=VS,LRECL=2052,RLKSIZ=2056)
//DATA. FT04F001 DD UNIT=(SYS0A,SEP=(FT01F001,FT03F001)),
// SPACE=(360,100),
// DCB=(RECFM=VS,LRECL=360,RLKSIZ=364)
//DATA. FT08F001 DD UNIT=2400,VOL=SER=6P1Z,DD,DS=01F01APE,
// DCB=(RECFM=VS,LRECL=2052,RLKSIZ=2056)
//DATA. FT10F001 DD UNIT=2400,VOL=SER=6P1Z,DD,DS=ST1F01, LABEL=1,
// DCB=(RECFM=VS,LRECL=2052,RLKSIZ=2056)
//DATA. FT12F001 DD UNIT=(SYS0A,SEP=(FT03F001,FT04F001)),
// SPACE=(1804,120),DCB=(RECFM=VS,LRECL=1800,RLKSIZ=1804)
//DATA. FT14F001 DD DDONLY
//DATA. FT20F001 DD UNIT=SYS0A,SPACE=(CYL,20),DS=0201K1APE,
// DCB=(RECFM=VS,LRECL=2052,RLKSIZ=2056)
//DATA. FT22F001 DD UNIT=SYS0A,SPACE=(CYL,20)
//DATA. INPUT DD *

```

Reproduced from
best available copy.